(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
3 May 2001 (03.05.2001)

PCT

(10) International Publication Number
WO 01/31888 A1

(51) International Patent Classification[7]: H04L 29/06

(21) International Application Number: PCT/US00/41211

(22) International Filing Date: 18 October 2000 (18.10.2000)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
09/426,614 26 October 1999 (26.10.1999) US

(63) Related by continuation (CON) or continuation-in-part (CIP) to earlier application:
US 09/426,614 (CON)
Filed on 26 October 1999 (26.10.1999)

(71) Applicant *(for all designated States except US)*: **3COM CORPORATION** [US/US]; 5400 Bayfront Plaza, Santa Clara, CA 95052 (US).

(72) Inventors; and

(75) Inventors/Applicants *(for US only)*: **BORELLA, Michael, S.** [US/US]; 1208 Haverhill Circle, Naperville, IL 60563 (US). **GRABELSKY, David, A.** [US/US]; 3800 Lee Street, Skokie, IL 60076 (US).

(74) Agent: **LESAVICH, Stephen**; McDonnell Boehnen Hulbert & Berghoff, 32nd floor, 300 South Wacker Drive, Chicago, IL 60606 (US).

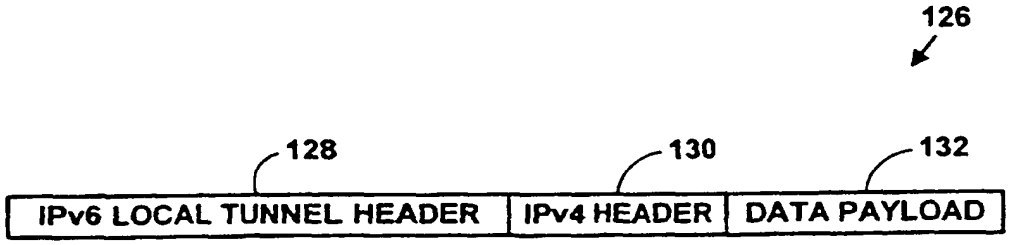
(81) Designated States *(national)*: AE, AG, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, BZ, CA, CH, CN, CR, CU, CZ, DE, DK, DM, DZ, EE, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MA, MD, MG, MK, MN, MW, MX, MZ, NO, NZ, PL, PT, RO, RU, SD, SE, SG, SI, SK, SL, TJ, TM, TR, TT, TZ, UA, UG, US, UZ, VN, YU, ZA, ZW.

(84) Designated States *(regional)*: ARIPO patent (GH, GM, KE, LS, MW, MZ, SD, SL, SZ, TZ, UG, ZW), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, GW, ML, MR, NE, SN, TD, TG).

Published:
— *With international search report.*
— *Before the expiration of the time limit for amending the claims and to be republished in the event of receipt of amendments.*

For two-letter codes and other abbreviations, refer to the "Guidance Notes on Codes and Abbreviations" appearing at the beginning of each regular issue of the PCT Gazette.

(54) Title: METHOD AND SYSTEM FOR DUAL-NETWORK ADDRESS UTILIZATION BY VIRTUAL TUNNELING



(57) Abstract: A method and system for dual network address utilization. The methods and system described herein may help the transition from Internet Protocol version-4 ("IPv4") networks to Internet Protocol version-6 ("IPv6") networks. Dual address allocation on IPv4 (50) and/or IPv6 (88) network addresses is provided via a dual protocol stack. Network devices may communicate with legacy IPv4 networks while using IPv6 network addresses on an IPv6 network for local communications. IPv6 (122) over IPv4 (120) remote virtual tunnels are used to allow network devices using IPv6 network addresses on a local IPv6 network to communicate with remote IPv6 networks over IPv4 public networks (e.g., the Internet). IPv4 (130) over IPv6 (128) local virtual tunnels may be used to allow network devices to using IPv4 network addresses on a local IPv6 network to communicate with remote IPv4 public networks. The IPv4 addresses allocated include IPv4 addresses that may be allocated and used for Distributed Network Address Translation ("DNAT") and/or the Realm Specific Internet Protocol ("RSIP"). The method and system can be used with virtually any set of networks that requires transitions between X-bit and Y-bit network addresses and dual network address utilization.

WO 01/31888 A1

METHOD AND SYSTEM FOR DUAL-NETWORK ADDRESS UTILIZATION BY VIRTUAL TUNNELING

FIELD OF INVENTION

This invention relates to computer networks. More specifically, it relates to a method and system for dual network address utilization.

BACKGROUND OF THE INVENTION

The Internet Protocol ("IP") is an addressing protocol designed to facilitate the routing of traffic within a network or between networks. The Internet Protocol is used on many computer networks including the Internet, intranets and other networks. Current versions of Internet Protocol such as Internet Protocol version-4 ("IPv4") are becoming obsolete because of limited address space. With a 32-bit address-field, it is possible to assign 2^{32} different addresses, which is 4,294,967,296, or greater than 4 billion globally unique addresses.

However, with the explosive growth of the Internet and intranets, Internet Protocol addresses using a 32-bit address-field may soon be exhausted. Internet Protocol version-6 ("IPv6") proposes the use of a 128-bit address-field for Internet Protocol addresses. However, a large number of legacy networks including a large number of Internet subnets will still be using older versions for Internet Protocol with a 32-bit address space for many years to come. As is known in the art, a subnet is smaller of part of a larger network using a similar network addressing scheme.

Network Address Translation ("NAT") has been proposed to extend the lifetime of Internet Protocol version 4 by allowing subnets with private Internet Protocol addresses to exist behind a single or small number of globally unique Internet Protocol addresses (see e.g., Internet Engineering Task Force ("ITEF") RFC 2663, "IP Network Address Translator ("NAT") Terminology and Considerations," by P. Srisuresh and M. Holdrege, August 1999). Each private host uses a single

global Internet Protocol address for communication with external networks such as the Internet.

Internally, a subnet may use local private addressing. Local private addressing may be any addressing scheme that is different from the public Internet Protocol addressing. The local addresses on a subnet are typically not available to the external, global Internet. When a device or node using local addressing desires to communicate with the external world, its local address is translated to a common external Internet Protocol address used for communication with an external network by a network address translation device. That is, network address translation allows one or more global Internet Protocol addresses to be shared among a larger number of network devices using local private addresses.

There are several problems associated with using network address translation to extend the life of the Internet Protocol version-4. Network address translation interferes with the end-to-end routing principal of the Internet that recommends that packets flow end-to-end between network devices without changing the contents of any packet along a transmission route (see e.g., "Routing in the Internet," by C. Huitema, Prentice Hall, 1995, ISBN 0-131-321-927).

Current versions of network address translation replace a local network address in a data packet header with an external global network address on outbound traffic, and replace an external global network address in a data packet header with a local private network address on inbound traffic. This type of address translation is computationally expensive, causes security problems by preventing certain types of encryption from being used, or breaks a number of existing applications in a network that cannot coexist with network address translation (e.g., File Transfer Protocol ("FTP")).

Current versions of network address translation may not gracefully scale beyond a small subnet containing a few dozen nodes or devices because of the computational and other resources required. Network address translation potentially requires support for many different application layer internal network protocols be specifically programmed into a translation mechanism such as a network address translation router.

Computational burdens placed on a network address translation router may be significant and degrade network performance, especially if several network address translation-enabled sub-networks share the same network address translation router. In a worst case scenario, a network address translation router translates every inbound and data packet.

Application Layer Gateways ("ALG") have also been used at a border between a private network and a public network like the Internet to provide address translation. As is known in the art, a gateway is a device that connects two networks using different communications protocols so that information can be passed from one to the other. A gateway both transfers information and converts it to a form compatible with the protocols used by a receiving network.

However, the Application Layer Gateways complicate the deployment of new applications. Sending and receiving systems need to support the new applications, and any Application Layer Gateways in a routing path must be able to identify new applications to provide network address translation.

Some of the problems associated with network address translation of private network addresses into public network addresses have been overcome with Distributed Network Address Translation ("DNAT") described in co-pending Applications No. 09/035,600, 09/270,967 and 09/271,025 assigned to the same

Assignee as the present application. See also "Distributed Network Address Translation", by Michael Borella, David Grabelsky, Ikhlaq Sidhu, and Brian Petry, IETF Internet Draft, <draft-borella-aatn-dnat-01.txt>, October 1998. Distributed Network Address Translation is also called "Realm Specific Internet Protocol" ("RSIP") by the IETF. For more information on Realm Specific Internet Protocol see "Realm Specific IP Framework," by M. Borella and J. Lo, IETF draft, <draft-ieft-nat-rsip-framework-02.txt>, October 1999, and "Realm Specific IP: Protocol Specification," by M. Borella and J. Lo, IETF draft, <draft-ietf-nat-rsip-protocol-02.txt>, August 1999.

For Distributed Network Address Translation or Realm Specific Internet Protocol, network devices request a set of locally unique ports from a Distributed Network Address Translation server or a Realm Specific Internet Protocol server for external communications with a public network like the Internet. A network device on a private network replaces default or ephemeral ports (e.g., such as Transmission Control Protocol or User Datagram Protocol) with the locally unique ports. The network device uses a combination network address including a locally unique port and a common external network address (e.g., an IP address) for the Distributed Network Address Translation server for communications with the external networks. The network devices use private network addresses for local communications on the private network.

A Distributed Network Address Translation server or a Realm Specific Internet Protocol server maintains a port-to-private network address table as locally unique ports are allocated to network devices. Network devices send data packets to external networks using a combination network address including a locally unique port and the common external network address via the Distributed Network Address

Translation server or Realm Specific Internet Protocol server. For inbound data packets from an external network, the Distributed Network Address Translation server or Realm Specific Internet Protocol uses the port-to-private network address table to route data packets back to the appropriate network device on the private network.

Distributed Network Address Translation or Realm Specific Internet Protocol allows a host to tunnel data packets to/from a network device and a server over a virtual tunnel. As is known in the art, a "virtual tunnel" is created by encapsulating a data packet inside another data packet. The outer header typically identifies the "endpoints" of the tunnel. The inner header typically identifies an original sender and recipient of the data. Thus, data packets are not modified between a source and a destination using Distributed Network Address Translation or Realm Specific Internet Protocol.

It is becoming commonplace for private stub network or subnets to be "multiple address networks." Multiple address networks are networks in which more than one type of network address is used. For example, a private subnet may use new 128-bit Internet Protocol version-6 addresses to communicate internally and may use the older 32-bit Internet Protocol version-4 addresses to communicate with external networks such as the Internet.

However, there are a number of problems associated with using Internet Protocol version-6 addresses on a private subnet and Internet Protocol version-4 addresses on public networks like the Internet. One problem is that Internet Protocol version-6 subnets and Internet Protocol version-4 subnets can not communicate directly with one another without translation of network addresses since Internet Protocol version-4 uses 32-bit addresses and Internet Protocol version-6 uses 128-bit

addresses. The network address translations required are subject to the network address translation problems described above.

Another problem is that some network devices will support only Internet Protocol version-6, others will support only Internet Protocol version-4, and still others will support both versions of the Internet Protocol. Network address translators have to be provided with information as to which network devices support which version of the Internet Protocol to provide network address translation. This complicates the deployment of new applications that are used across networks.

Thus, it is desirable to provide a solution that allows legacy Internet Protocol version-4 subnets to be connected to and communicate with newer Internet Protocol version-6 subnets. The solution should allow network devices to use any combination of Internet Protocol version-6 and/or Internet Protocol version-4 on a subnet with limited computational burdens and without complicating deployment of new applications.

SUMMARY OF THE INVENTION

In accordance with preferred embodiments of the present invention, some of the problems associated supporting legacy networks are overcome. A method and system for dual network address utilization is provided.

One aspect of the invention includes a method for dual network address utilization. A dual protocol stack provides dual address allocation of X-bit and Y-bit network addresses (e.g., 128-bit Internet Protocol version-6 and 32-bit Internet Protocol version-4 network addresses). Network devices communicate with legacy Y-bit networks while using X-bit network addresses on an X-bit network for local communications. X-bit over Y-bit remote virtual tunnels are used to allow network devices using X-bit network addresses on a local X-bit network to communicate with remote X-bit networks over Y-bit networks (e.g., the Internet). Y-bit over X-bit local virtual tunnels may be used to allow network devices to using Y-bit network addresses on a local X-bit network to communicate with remote Y-bit networks. The Y-bit addresses allocated include Y-bit addresses may also used for the Distributed Network Address Translation protocol and/or the Realm Specific Internet Protocol.

Another aspect of the invention includes a system for dual network address utilization. The dual network address system includes a multiple network devices including a dual protocol stack and a virtual tunnel gateway. The dual protocol stack includes a first portion for networking protocols using X-bit network addresses and a second portion for networking protocols using Y-bit network addresses. The virtual tunnel gateway is used for adding a remote tunnel header for a remote virtual tunnel for a data packet with a header including X-bit network addresses sent from a local network by a local network device using X-bit network addresses across an

intermediate network using Y-bit network addresses to a remote network device on a remote network using X-bit network addresses. The virtual tunnel gateway is also used for removing a local tunnel header including X-bit network addresses for a data packet with a header including Y-bit network addresses and for transmitting the data packet with the header including Y-bit network address across the intermediate network using Y-bit network addresses.

The methods and system described herein may help the transition from Internet Protocol version-4 ("IPv4") networks to Internet Protocol version-6 ("IPv6") networks. However, the present invention is not limited to such an embodiment, and can be used with virtually any set of networks that require transitions between X-bit and Y-bit network addresses and dual network address utilization.

The foregoing and other features and advantages of a preferred embodiment of the present invention will be more readily apparent from the following detailed description. The detailed description proceeds with references to the accompanying drawings.

BRIEF DESCRIPTION OF THE DRAWINGS

Preferred embodiments of the present inventions are described with reference to the following drawings, wherein:

FIG. 1 is a block diagram illustrating an exemplary network system;

FIG. 2 is a block diagram illustrating a protocol stack for a network device;

FIG. 3 is a block diagram illustrating an exemplary dual network address utilization system;

FIG. 4 is a block diagram illustrating an exemplary dual protocol stack;

FIGS. 5A and 5B are a flow diagram illustrating a method for dual network address utilization;

FIG. 6 is a block diagram illustrating an exemplary X-bit to Y-bit network address data packet; and

FIG. 7 is a block diagram illustrating an exemplary Y-bit to X-bit network address data packet.

DETAILED DESCRIPTION OF PREFERRED EMBODIMENTS

Exemplary network system

FIG. 1 is a block diagram illustrating an exemplary network system 10 for one preferred embodiment of the present invention. The network system 10 includes a first computer network 12 with multiple network devices (14, 16, 18, 20, 22) and a router 26 to route data packets to another external computer network. The multiple network devices include any of computers (14, 18), printers 16, personal digital assistants 20, telephones 22, or other hand-held devices or other network devices that can be connected to the first computer network 12. The first computer network 12 may also include a Dynamic Host Configuration Protocol ("DHCP") server 24. As is known in the art, the Dynamic Host Configuration Protocol is a protocol for dynamically passing network addresses such as Internet Protocol addresses and configuration information to network devices on a network. For more information on DHCP see IETF RFC-1541, and RFC-2131 and RFC-2132, incorporated herein by reference.

The first computer network 12 has an external common network address 28 (e.g., a global Internet Protocol version-4 address, 198.10.20.30) to identify the first computer network 12 to an external computer network such as a second computer network 30 and/or a third computer network 32 external to the first private computer network-x 12. The multiple network devices (14, 16, 18, 20, 22, 24 and 26) have a local network address (e.g., an Internet Protocol version-6) on the first computer network 12. In one preferred embodiment of the present invention, a network access service provider 34 with a router 36 routes data packets to/from the first computer network 12 a to second computer network 30 and/or to a third computer network 32 through a second network switch 38 and/or a third network switch 40. In another

embodiment of the present invention, the first computer network 12 is connected directly to the second computer network 30. The first computer network 12 is also connected to a second computer network 42 via computer networks 30 or 32. The second computer network 42 is also a computer network that includes multiple network devices (not illustrated in FIG. 1) that use local internal network addresses (e.g., Internet Protocol version-6 addresses) behind a public globally routable network address of (e.g., a global Internet Protocol version-4 address 192.200.20.3).

In one preferred embodiment of the present invention, the first computer network 12 is a Small Office/Home Office ("SOHO") Local Area Network ("LAN"), also called a "legacy" LAN. The first computer network 12 can also be a "stub" network or a sub-network ("subnet"). As is known in the art, a "stub" network is an end or terminal network. As is known in the art, a "subnet" is a smaller part of a larger network that uses a common addressing scheme (e.g., Internet Protocol addresses). The second network 30 is the Internet or an intranet, using Internet Protocol version-4 network addresses and the third network 32 is a Public Switched Telephone Network ("PSTN"). The second network-y 42 is also a SOHO LAN, stub network or subnet. However, other network types and network components can also be used and the present invention is not limited to the network types and network components described for this preferred embodiment.

Network devices and routers for preferred embodiments of the present invention include network devices that can interact with network system 10 and network system 68 (FIG. 3) discussed below that are compliant with all or part of standards proposed by the Institute of Electrical and Electronic Engineers ("IEEE"), International Telecommunications Union-Telecommunication Standardization Sector ("ITU"), Internet Engineering Task Force ("IETF"), the Wireless Application

Protocol ("WAP") Forum, and Data-Over-Cable-Service-Interface-Specification ("DOCSIS") standards for Multimedia Cable Network Systems ("MCNS"). However, network devices based on other standards could also be used. IEEE standards can be found on the World Wide Web at the Universal Resource Locator ("URL") "www.ieee.org." The ITU, (formerly known as the CCITT) standards can be found at the URL "www.itu.ch." IETF standards can be found at the URL "www.ietf.org." The WAP standards can be found at the URL "www.wapforum.org." The DOCSIS standards can be found at the URL "www.cablemodem.com."

An operating environment for network devices and routers of the present invention include a processing system with at least one high speed Central Processing Unit ("CPU") and a memory. In accordance with the practices of persons skilled in the art of computer programming, the present invention is described below with reference to acts and symbolic representations of operations or instructions that are performed by the processing system, unless indicated otherwise. Such acts and operations or instructions are referred to as being "computer-executed" or "CPU executed."

It will be appreciated that acts and symbolically represented operations or instructions include the manipulation of electrical signals by the CPU. An electrical system represents data bits which cause a resulting transformation or reduction of the electrical signals, and the maintenance of data bits at memory locations in a memory system to thereby reconfigure or otherwise alter the CPU's operation, as well as other processing of signals. The memory locations where data bits are maintained are physical locations that have particular electrical, magnetic, optical, or organic properties corresponding to the data bits.

The data bits may also be maintained on a computer readable medium including magnetic disks, optical disks, organic memory, and any other volatile (e.g., Random Access Memory ("RAM")) or non-volatile (e.g., Read-Only Memory ("ROM")) mass storage system readable by the CPU. The computer readable medium includes cooperating or interconnected computer readable medium, which exist exclusively on the processing system or be distributed among multiple interconnected processing systems that may be local or remote to the processing system.

Exemplary protocol stack

FIG. 2 is a block diagram illustrating an exemplary layered protocol stack 44 for network devices on Internet Protocol version-4 subnets on the exemplary dual stack network system 68 (FIG. 3) discussed below. The layered protocol stack 44 is described with respect to Internet Protocol suites comprising from lowest-to-highest, a link, network, transport and application layer. However, more or fewer layers could also be used, and different layer designations could also be used for the layers in the protocol stack 44 (e.g., layering based on the Open Systems Interconnection ("OSI") model).

The network devices are connected to a computer network with Network Interface Card ("NIC") device drivers in a link layer 46 for the hardware network devices connecting the network devices to the computer network 12. The link layer 44 may include a Medium Access Control ("MAC") protocol layer or other data-link layer protocol.

Above the link layer 48 is a network layer 48 (also called the Internet Layer for Internet Protocol suites). The network layer 48 includes an Internet Protocol version-4 ("IPv4") layer 50. As is known in the art, IPv4 50 is an addressing protocol designed to route traffic within a network or between networks. IPv4 layer 50,

hereinafter IPv4 50, is described in IETF RFC-791, incorporated herein by reference. The network layer 48 also includes an Internet Group Management Protocol version-4 ("IGMPv4") layer 52, and an optional Internet Control Message Protocol version-4 ("ICMPv4") layer 54. The IGMP v4 layer 52 and the ICMP v4 layer 54 are used with IPv4 50.

ICMPv4 layer 52, hereinafter ICMPv4 52, is used for Internet Protocol control. The main functions of ICMPv4 52 include error reporting, reachability testing (e.g., "pinging"), route-change notification, performance, subnet addressing and other maintenance. For more information on ICMPv4 52 see IETF RFC-792, incorporated herein by reference. IGMPv4 layer 54, hereinafter IGMPv4 54, is responsible for multicasting. For more information on IGMPv4 54 see IETF RFC-1112, incorporated herein by reference. IGMPv4 54 are not both required in the protocol stack 44. ICMP v4 52 is often used without IGMPv4 54.

The network layer 48 may also include an optional Distributed Network Address Translation ("DNAT") or Realm Specific Internet Protocol ("RSIP") layer 55. The DNAT/RSIP layer 55 is used to allocate locally unique ports as well as a combination network address including a locally unique port and a common external network address (e.g., an IP v4 address) for a DNAT/RSIP server for communications with the external networks. For more information on DNAT, see co-pending U.S. Application No. 09/035,600 incorporated herein by reference. DNAT is also referred to as "Realm Specific Internet Protocol" ("RSIP") by the IETF. For more information on RSIP see, "Realm Specific IP Framework," by M. Borella and J. Lo, IETF draft, <draft-ieft-nat-rsip-framework-02.txt>, October 1999, "Realm Specific IP: Protocol Specification," by M. Borella and J. Lo, IETF draft, <draft-ietf-nat-rsip-protocol-02.txt>, August 1999, incorporated herein by reference.

Above network layer 48 is a transport layer 56. The transport layer 56 includes a Transmission Control Protocol ("TCP") layer 58 a User Datagram Protocol ("UDP") layer 60, and an optional DNAT/RSIP layer 55 described above. The TCP layer 58, hereinafter TCP 58, provides a connection-oriented, end-to-end reliable protocol designed to fit into a layered hierarchy of protocols which support multi-network applications. TCP 58 provides for reliable inter-process communication between pairs of processes in network devices attached to distinct but interconnected networks. For more information on TCP 58 see IETF RFC-793, incorporated herein by reference.

The UDP layer 60, hereinafter UDP 60, provides a connectionless mode of communications with datagrams in an interconnected set of computer networks. UDP 60 provides a transaction oriented datagram protocol, where delivery and duplicate packet protection are not guaranteed. For more information on UDP 60 see IETF RFC-768, incorporated herein by reference. Both TCP 58 and UDP 60 are not required in protocol stack 42. Either TCP 58 or UDP 60 can be used without the other.

Above the transport layer 56 is an application layer 62 including application programs 64. The application programs 64 provide desired functionality to a network device (e.g., telephony or other communications functionality).

Exemplary dual network address utilization system

FIG. 3 is a block diagram illustrating an exemplary dual network address utilization system 68. The exemplary network utilization system 68 includes an IPv4 Internet backbone 70, one or more IPv4 only subnets 72, one or more IPv6 only subnets 74, and one or more IPv4/IPv6 dual protocol stack subnets 76 and 76' (e.g., first network-x 12 or second network-y 42). FIG. 3 illustrates only one IPv4 only

subnet 72, only one IPv6 only subnet 74 and two IPv4/IPv6 dual protocol stack subnets 74 and 76 for the sake of simplicity. However, the exemplary dual network address utilization system 68 typically may include tens or hundreds of such subnets. Exemplary IPv4/IPv6 dual protocol stack subnet 76 includes an exemplary first network device 78 (e.g., a computer) and an exemplary first virtual tunnel gateway 80. Exemplary IPv4/IPv6 dual protocol stack subnet 76' includes an exemplary second network device 82 and an exemplary second virtual tunnel gateway 84. Use of network devices 78, 82 including a dual protocol stack 86 as is explained below.

The virtual tunnel gateways 80, 84 are used to create a virtual tunnel. The virtual tunnel gateways 80, 84 may also include a DNAT/RSIP server. As is known in the art, a gateway is a device that connects networks using different communications protocols so that information can be passed from one to the other. A gateway both transfers information and converts it to a form compatible with the protocols used by the receiving network.

As is known in the art, a "virtual tunnel" can be created by encapsulating a data packet inside another data packet. For example, an outer header is added before an inner header of a data packet. Between the inner header and outer headers are any other headers for a data path, or security, such as security headers specific to a tunnel configuration. The outer header typically identifies the "endpoints" of the tunnel. The inner header typically identifies an original sender and recipient of the data. For more information, see "IP-in-IP tunneling," by W. Simpson, IETF RFC-1853, October 1995, incorporated herein by reference. However, the present invention is not limited this exemplary architecture and more or fewer, and other types of network devices and subnet may also be used.

In the exemplary network utilization system 68, the IPv4 Internet backbone 70 connects a number of legacy IPv4 only subnets 72, new IPv6 only subnets 74 and new dual protocol stack IPv4/IPv6 subnets 76. Communication over the IPv4 Internet backbone between any two subnet uses IPv4 or uses an IPv4 virtual tunnel. Note that the IPv4 only subnet 72 and the IPv6 only subnet 74 cannot currently communicate directly with each other without some form of direct IPv4-to-IPv6 and IPv6-to-IPv4 network address translation.

Exemplary dual protocol stack

FIG. 4 is a block diagram illustrating an exemplary dual protocol stack 86. The dual protocol stack 86 is used on network devices and virtual tunnel gateways on dual protocol stack IPv4/IPv6 subnets 76. The dual protocol stack 86 includes a link layer 46, a network layer 48, a transport layer 56 and an application layer 62 with the protocols described above for the protocol stack 44 illustrated in FIG. 2.

In addition, the dual protocol stack 86 includes a first IP layer including 32-bit versions of IPv4 50, ICMPv4 52. The dual protocol stack 74 also includes a second IP layer including 128-bit versions of IPv6 88, and ICMPv6 90. IPv6 88 now includes IGMP functionality so, no separate IGMPv6 layer is illustrated. The dual protocol stack 86 can send and receive IP data packets with either 32-bit IPv4 addresses and/or 128-bit IPv6 addresses.

For more information on IPv6 88 see IETF-RFC 2460, "Internet Protocol, Version 6 ("IPv6") Specification," by S. Deering and R. Hinden, December 1998, incorporated herein by reference. For more information on ICMPv6 90 see IETF-RFC 2463, "Internet Control Message Protocol ("ICMPv6") for the Internet Protocol

Version 6 ("IPv6") Specification", by A. Conta and S. Deering, December 1998, incorporated herein by reference.

Network devices and virtual tunnel gateways include dual protocol stack 86 on dual protocol stack IPv4/IPv6 subnets 76. Network devices on IPv4 only subnets 72 include protocol stack 44. Network devices on IPv6 subnets 74 only include a protocol stack with versions of IPv6 protocols (not illustrated in the FIGs.).

Dual network address utilization

FIGS. 5A and 5B are a flow diagram illustrating a Method 94 for dual network address utilization. In FIG. 5A at Step 96, a data packet is received in a protocol stack on a dual protocol stack on a first network device on a local first network for a second network device on a remote second network. The local first network uses X-bit network addresses and the remote second network uses X-bit network addresses. The local first network is connected to the remote second network by a third network that uses Y-bit network addresses. At Step 98, a test is conducted from the protocol stack on the dual protocol stack on the first network device to determine whether the data packet will use an X-bit network address.

If the data packet will use an X-bit network address, at Step 100 a first portion of the dual protocol stack is selected with a networking protocol using X-bit network addresses. At step 102, a header is added to the data packet including X-bit network addresses (e.g., X-bit source and destination network addresses). At Step 104, the data packet is transmitted to a local virtual tunnel gateway on the local first network. The local virtual tunnel gateway adds a virtual tunnel header with a Y-bit network addresses to the data packet and transmits the data packet to the second network device on the remote second network over a remote virtual tunnel via the third network that uses Y-bit network addresses.

A remote virtual tunnel gateway on the remote second network receives the data packet including the tunnel header with the Y-bit network addresses. The remote virtual tunnel header removes the tunnel header. The data packet is transmitted to the second network device using the X-bit network addresses in the data packet since the remote second network uses X-bit network addresses for data packets.

If the data packet will use is not an X-bit network address at Step 98 of FIG. 5A, at Step 106 of FIG. 5B a second portion of the dual protocol stack is selected with networking protocols using Y-bit network addresses. At step 108 a header is added to the data packet including Y-bit network addresses (e.g., Y-bit source and destination addresses). Step 108 may also include allocating a Y-bit DNAT/RSIP address that is added to the header. At Step 110, a virtual tunnel header with a X-bit addresses is added to the data packet. At Step 112, the data packet is transmitted to a local virtual tunnel gateway via a local virtual tunnel using the X-bit network addresses in the virtual tunnel header. At Step 114 the X-bit tunnel header is removed on the local virtual tunnel gateway. At Step 116, the data packet is transmitted from the local virtual tunnel gateway on the local first network to the remote second network over the third network using the Y-bit network addresses in the data packet.

The remote virtual tunnel gateway on the remote second network receives the data packet with Y-bit network addresses. The remote virtual gateway adds a tunnel header with X-bit network addresses. The data packet is transmitted to the second network device on the remote second network using the X-bit network addresses since the remote second network uses X-bit network addresses for data packets and the original data packet includes Y-bit network addresses.

In one exemplary preferred embodiment of the present invention Method 94 is used with exemplary dual utilization system network 68. However, the present

invention is not limited to such an embodiment, and other embodiments can also be used.

In such an exemplary embodiment in FIG. 5A at Step 96, a data packet is received on a dual protocol stack 86 on a first network device 78 on a IPv4/IPv6 dual stack subnet 76 for a second network device 82 on a remote second network. The remote second network is a IPv6 only subnet 74 or IPv4/IPv6 dual stack subnet 76'. The exemplary embodiment is illustrated only with respect to the remote IPv4/IPv6 dual stack subnet 76'. However, the remote second network can also be the IPv6 only subnet 74. The local IPv4/IPv6 subnet 76 uses 128-bit IPv6 addresses for local network addresses. The remote IPv4/IPv6 subnet 76' also uses 128-bit IPv6 addresses for local network addresses. The local dual stack IPv4/IPv6 subnet 76 is connected to the remote IPv4/IPv6 subnet 76' by an IPv4 Internet backbone 70 that uses 32-bit IPv4 addresses.

At Step 98, a test is conducted from the dual protocol stack 86 to determine whether the data packet will use 128-bit IPv6 addresses. If the data packet will use 128-bit IPv6 addresses, at Step 100 a first portion of the dual protocol stack 86 with IPv6 protocols (e.g., 88, 90, 92) is selected. At step 102, a header is added to the data packet including 128-bit IPv6 source and destination addresses. At Step 104, the data packet with 128-bit IPv6 addresses is transmitted to a local virtual tunnel gateway 80 on the local dual stack IPv4/IPv6 subnet 76 using the 128-bit IPv6 addresses. The local virtual tunnel gateway 80 adds a virtual tunnel header with 32-bit IPv4 source and destination addresses by calling the IPv4 portion of the dual protocol stack 86. The virtual tunnel gateway 80 transmits the data packet to the second network device 82 on the remote IPv4/IPv6 subnet 76' over a remote virtual tunnel via the IPv4 Internet backbone 70 that uses 32-bit IPv4 addresses. On the remote IPv4/IPv6 subnet

76', a remote virtual tunnel gateway 84 removes the 32-bit IPv4 tunnel header and transmits the data packet to the remote second network device 82 using the original 128-bit IPv6 network addresses.

FIG. 6 is a block diagram illustrating an exemplary X-bit to Y-bit network address data packet 118 transmitted at Step 104. The data packet 118 includes a remote virtual tunnel header 120 with a 32-bit IPv4 addresses, a header 122 with 128-bit IPv6 addresses and a data payload 124.

If the data packet will not use an X-bit network address at Step 98 of FIG. 5A, at Step 106 of FIG. 5B a second portion of the dual protocol stack 86 with IPv4 protocols (e.g., 50, 52, 54) is selected. In one exemplary preferred embodiment of the present invention, the 32-bit IPv4 address is a "normal" IPv4 address. In another exemplary preferred embodiment of the present invention, the 32-bit IPv4 address is a DNAT/RSIP IPv4 combination network address.

As was discussed above, the DNAT/RSIP combination network address includes a locally unique port and a common external network address (e.g., an IPv4 address) used for DNAT/RSIP. The DNAT/RSIP IPv4 combination network address allows a subnet to use a larger number of private network addresses behind a smaller number of public network addresses. Step 108 may also include allocating 32-bit IPv4 DNAT/RSIP addresses that are added to the header.

At step 108, a header is added to the data packet including a 32-bit IPv4 source and destination network address. At Step 110, a local virtual tunnel header with a 128-bit IPv6 source and destination addresses is added to the data packet from the IPv4 portion of the dual protocol stack 86 by calling the IPv6 portion of the dual protocol stack 86. At Step 112, the data packet is transmitted to the local virtual tunnel gateway 80 via a local virtual tunnel using the 128-bit IPv6 addresses in the

virtual tunnel header since the local IPv4/IPv6 subnet 76 uses 128-bit IPv6 addresses. At Step 114, the 128-bit IPv6 tunnel header is removed on the local virtual tunnel gateway 80. At Step 116, the data packet is transmitted from the virtual tunnel gateway 80 to the second network device 82 on remote second network 76' over the IPv4 Internet backbone using the 32-bit IPv4 addresses in the data packet.

On the remote IPv6/IPv6 subnet 76', the remote virtual tunnel gateway 84 adds a tunnel header with a 128-bit IPv6 source and destination address to the data packet and transmits the data packet to the second network device 82 since the remote IPv4/IPv6 subnet 76' uses 128-bit network addresses for data packets and the original data packet includes a 32-bit IPv4 network address. The remote virtual tunnel gateway 84 may also use DNAT/RSIP on the data packet to identify the second network device and route the data packet to the second network device.

FIG. 7 is a block diagram illustrating an exemplary Y-bit to X-bit network address data packet 126 transmitted at Step 104. The data packet 126 includes a remote virtual tunnel header 128 with a 128-bit IPv6 addresses, a header 130 with 32-bit IPv4 addresses and a data payload 132.

The methods and system described herein may help the transition from IPv4 to IPv6 networks. Dual address allocation is provided with a dual protocol stack that allows network devices to communicate with legacy IPv4 networks while using IPv6 for local communications on a IPv6 subnet. IPv6 over IPv4 remote virtual tunnels may be used to allow network devices using IPv6 network addresses on a local IPv6 subnet to communicate with remote IPv6 subnets over IPv4 public subnets. IPv4 over IPv6 local virtual tunnels may be used to allow network devices to using IPv4 network addresses on a local IPv6 subnet to communicate with remote IPv4 public subnets. The IPv4 addresses allocated may include IPv4 addresses used for the

Distributed Network Address protocol Translation and/or the Realm Specific Internet Protocol.

It should be understood that the programs, processes, methods, systems and/or apparatus described herein are not related or limited to any particular type of computer apparatus (hardware or software), unless indicated otherwise. Various types of general purpose or specialized computer apparatus may be used with or perform operations in accordance with the teachings described herein. While various elements of the preferred embodiments have been described as being implemented in software, in other embodiments hardware or firmware implementations may alternatively be used and visa-versa.

In view of the wide variety of embodiments to which the principles of the invention can be applied, it should be understood that the illustrative embodiments are exemplary only, and should not be taken as limiting the scope of the present invention. For example, the steps of the flow diagrams may be taken in sequences other than those described, and more or fewer elements or component may be used in the block diagrams.

The claims should not be read as limited to the described order or elements unless stated to that effect. Therefore, all embodiments that come within the scope and spirit of the following claims and equivalents thereto are claimed as the invention.

WE CLAIM:

1. A method for dual network address utilization, comprising the steps of:

receiving an data packet in a dual protocol stack on a first network device on a local first network for a second network device on a remote second network, wherein the local first network uses X-bit network addresses, the remote second network uses X-bit network addresses and the local first network is connected to the remote second network by a third network that uses Y-bit network addresses;

determining from the dual protocol stack whether the data packet will use an X-bit network address, and if so,

selecting a first portion of the dual protocol stack including networking protocols using X-bit network addresses,

adding a header with an X-bit network addresses to the data packet,

transmitting the data packet to a local virtual tunnel gateway on the local first network using the X-bit network addresses, wherein the local virtual tunnel gateway adds a remote virtual tunnel header with Y-bit network addresses and transmits the data packet to the second network device on the remote second network over a remote virtual tunnel via the third network that uses Y-bit network addresses;

determining from the dual protocol stack whether the data packet will use an X-bit network addresses, and if not,

selecting a second portion of the dual protocol stack including networking protocols using Y-bit network addresses,

adding a header with a Y-bit addresses to the data packet,

adding a local virtual tunnel header with X-bit network addresses to the data packet,

transmitting the data packet to the virtual tunnel gateway over a local virtual tunnel using the X-bit network addresses in the virtual tunnel header,

removing the X-bit tunnel header on the local virtual tunnel gateway, and

transmitting the data packet from the local virtual tunnel gateway on the local first network to the second network device on the remote second network over the third network using the Y-bit network addresses in the data packet.

2. A computer readable medium have stored therein instructions for causing a central processing unit to execute the method of Claim 1.

3. The method of Claim 1 wherein the local first network and the remote second network include any of Internet Protocol version-6 subnets or dual Internet Protocol version-4/Internet Protocol version-6 subnets.

4. The method of Claim 1 wherein the third network includes an Internet Protocol version-4 network.

5. The method of Claim 4 wherein the third network is the Internet.

6. The method of Claim 1 wherein the X-bit network address includes a 128-bit Internet Protocol version-6 network address and the Y-bit network address includes a 32-bit Internet Protocol version-4 network address.

7. The method of Claim 1 wherein the first portion of the protocol stack includes an Internet Protocol version-6 portion of the protocol stack and the second portion of the protocol stack is an Internet Protocol version-4 portion of the protocol stack.

8. The method of Claim 1 wherein the Y-bit network address includes any of a Y-bit Distributed Network Address Translation or a Realm Specific Internet Protocol network address.

9. The method of Claim 8 wherein the Y-bit network address includes a combination network address including a common public Internet Protocol version-4 address and a locally unique port used for Distributed Network Address Translation or Realm Specific Internet Protocol to uniquely identify the first network device.

10. The method of Claim 1 wherein the first network device and the second network device include any of a computer, printer, personal digital assistant or telephone.

11. The method of Claim 1 further comprising:

receiving a data packet on a remote virtual tunnel gateway on the remote second network with a virtual tunnel header including Y-bit network addresses;

removing the virtual tunnel header on the remote virtual tunnel gateway;

transmitting the data packet to the second network device using X-bit network addresses from a header in the data packet, wherein the remote second network uses X-bit network addresses for data packets.

12. The method of Claim 1 further comprising:

receiving the data packet with a header including Y-bit network addresses on a remote virtual tunnel gateway on the remote second network;

adding a virtual tunnel header with X-bit network addresses to the data packet on the remote virtual tunnel gateway;

transmitting the data packet to the second network device using the X-bit network addresses in the tunnel header, wherein the remote second network uses X-bit network addresses for data packets and the data packet includes a header with Y-bit network addresses.

13. The method of Claim 1 wherein the networking protocols of the dual protocol stack include any of Internet Protocol version-4 or Internet Protocol version-6.

14. A dual network address utilization system, comprising in combination:

a dual protocol stack, wherein the dual protocol stack includes a first portion for networking protocols using X-bit network addresses and a second portion for networking protocols using Y-bit network addresses;

a plurality of network devices including the dual protocol stack;

a local network using X-bit network addresses;

a remote network using X-bit network addresses;

an intermediate network using Y-bit network addresses;

a virtual tunnel gateway for adding a remote tunnel header for a remote virtual tunnel for a data packet with a header including X-bit network addresses sent from a local network device on a local network using X-bit network addresses across an intermediate network using Y-bit network addresses to a remote network device on a remote network using X-bit network addresses for removing a local tunnel header including X-bit network addresses for a data packet with a header including Y-bit network addresses, and for transmitting the data packet with the header including Y-bit network address across the intermediate network using Y-bit network addresses.

15. The system of claim 14 wherein the X-bit network address is a 128-bit Internet Protocol version-6 network address and the Y-bit network address is a 32-bit Internet Protocol version-4 network address.

16. The system of Claim 14 wherein the Y-bit network address includes any of a Y-bit Distributed Network Address Translation or a Realm Specific Internet Protocol network address.

17. The system of Claim 16 wherein the Y-bit network address includes a combination network address including a common public Internet Protocol version-4 address and a locally unique port used for Distributed Network Address Translation or Realm Specific Internet Protocol to uniquely identify the first network device.

18. The system of Claim 14 wherein the plurality of network devices include any of a computers, printers, personal digital assistants or telephones.

19. A method for dual network address utilization, comprising the steps of:

receiving an data packet in a dual Internet Protocol stack on a first network device on a local Internet Protocol network for a second network device on a remote Internet Protocol network, wherein the local Internet Protocol network uses 128-bit Internet Protocol version-6 addresses, the remote second network uses 128-bit Internet Protocol Network addresses and the local Internet Protocol network is connected to the remote Internet Protocol network by a third Internet Protocol network that uses 32-bit Internet Protocol addresses;

determining from the dual Internet Protocol stack whether the data packet will use a 128-bit Internet Protocol version-6 address, and if so,

selecting a first portion of the dual Internet Protocol stack using 128-bit Internet Protocol version-6 addresses,

adding a header to the data packet including a 128-bit Internet Protocol version-6 network addresses,

transmitting the data packet to a local virtual tunnel gateway on the local Internet Protocol network using the header with 128-bit Internet Protocol version-6 network addresses, wherein the local virtual tunnel gateway adds a remote virtual tunnel header including 32-bit Internet Protocol version-4 network addresses and transmits the data packet to the second network device on the remote Internet Protocol network over a remote virtual tunnel via the third Internet Protocol network that uses 32-bit Internet Protocol version-4 addresses;

determining from the dual Internet Protocol stack whether the data packet will use a 128-bit Internet Protocol version-6 address, and if not,

selecting a second portion of the dual Internet Protocol stack using 32-bit Internet Protocol version-4 addresses,

adding a header to the data packet including 32-bit Internet Protocol version-4 network addresses,

adding a virtual tunnel header to the data packet including 128-bit Internet Protocol version-6 network addresses,

transmitting the data packet in a local virtual tunnel to the local virtual tunnel gateway using the 128-bit Internet Protocol addresses version-6 in the virtual tunnel header,

removing the virtual tunnel header with the 128-bit Internet Protocol addresses version-6 on the local virtual tunnel gateway, and

transmitting the data packet from the local virtual tunnel gateway on the local Internet Protocol network to the second network device on the remote Internet Protocol network over the third Internet Protocol network using the 32-bit Internet Protocol version-4 addresses in the data packet.

20. A computer readable medium have stored therein instructions for causing a central processing unit to execute the method of Claim 19.

21. The method of Claim 19 wherein the third Internet Protocol Network is the Internet.

22. The method of Claim 19 wherein the local Internet Protocol network and the remote Internet Protocol network include dual Internet Protocol version-4/Internet Protocol version-6 subnets.

23. The method of Claim 19 wherein the 32-bit Internet Protocol network address includes any of a 32-bit Internet Protocol version-4 Distributed Network Address Translation network address or a Realm Specific Internet Protocol network address.

24. The method of Claim 23 wherein the 32-bit Internet Protocol network address includes a combination network address with a common public Internet Protocol version-4 address and a locally unique port used for Distributed Network Address Translation or Realm Specific Internet Protocol to uniquely identify the first network device.

FIG. 1

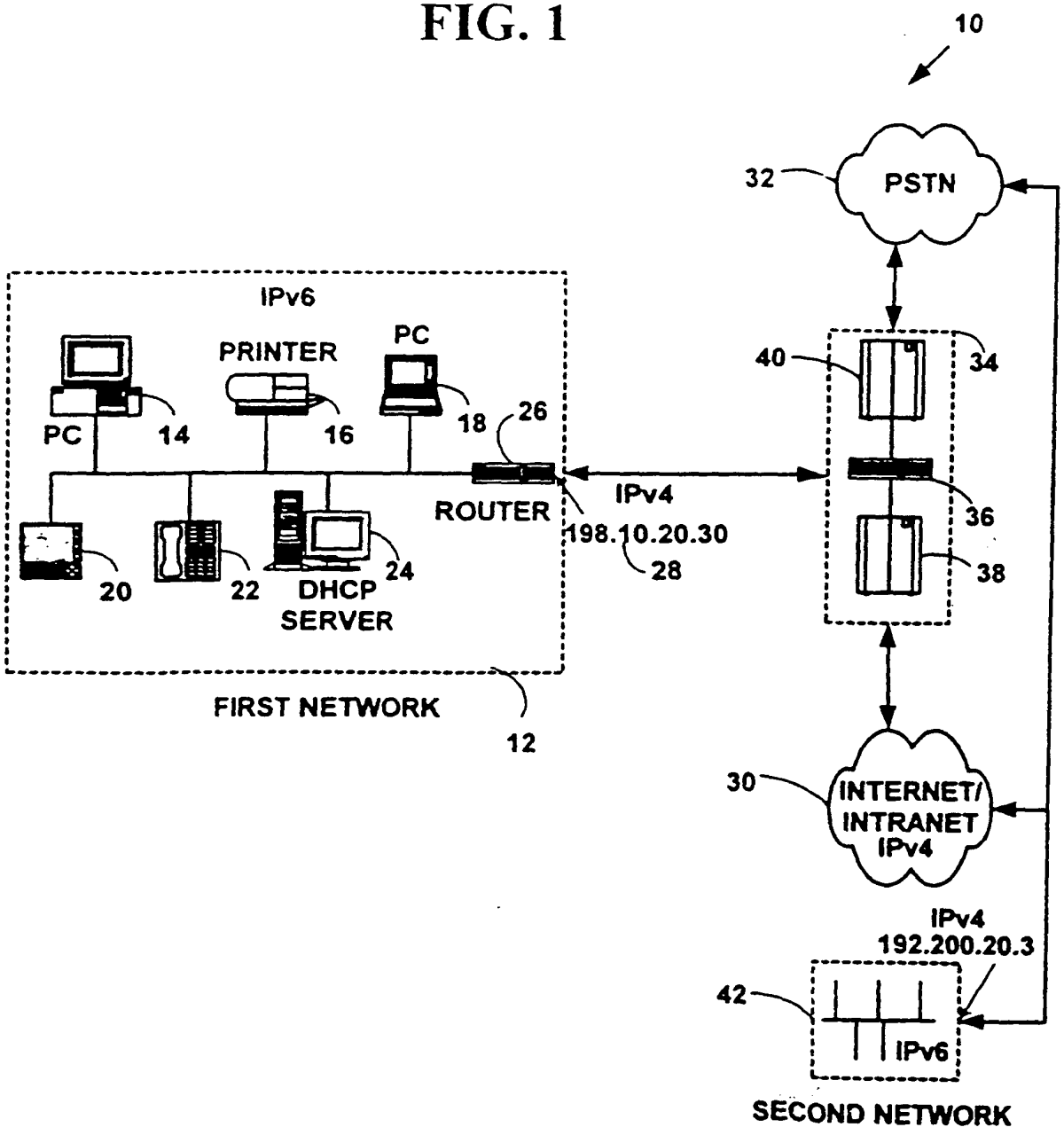

10
32
PSTN
IPv6
PRINTER
PC
PC
14
16
18
26
40
34
ROUTER
IPv4
198.10.20.30
28
36
38
20
22
DHCP
SERVER
24
FIRST NETWORK
12
30
INTERNET/
INTRANET
IPv4
IPv4
192.200.20.3
42
IPv6
SECOND NETWORK

FIG. 2

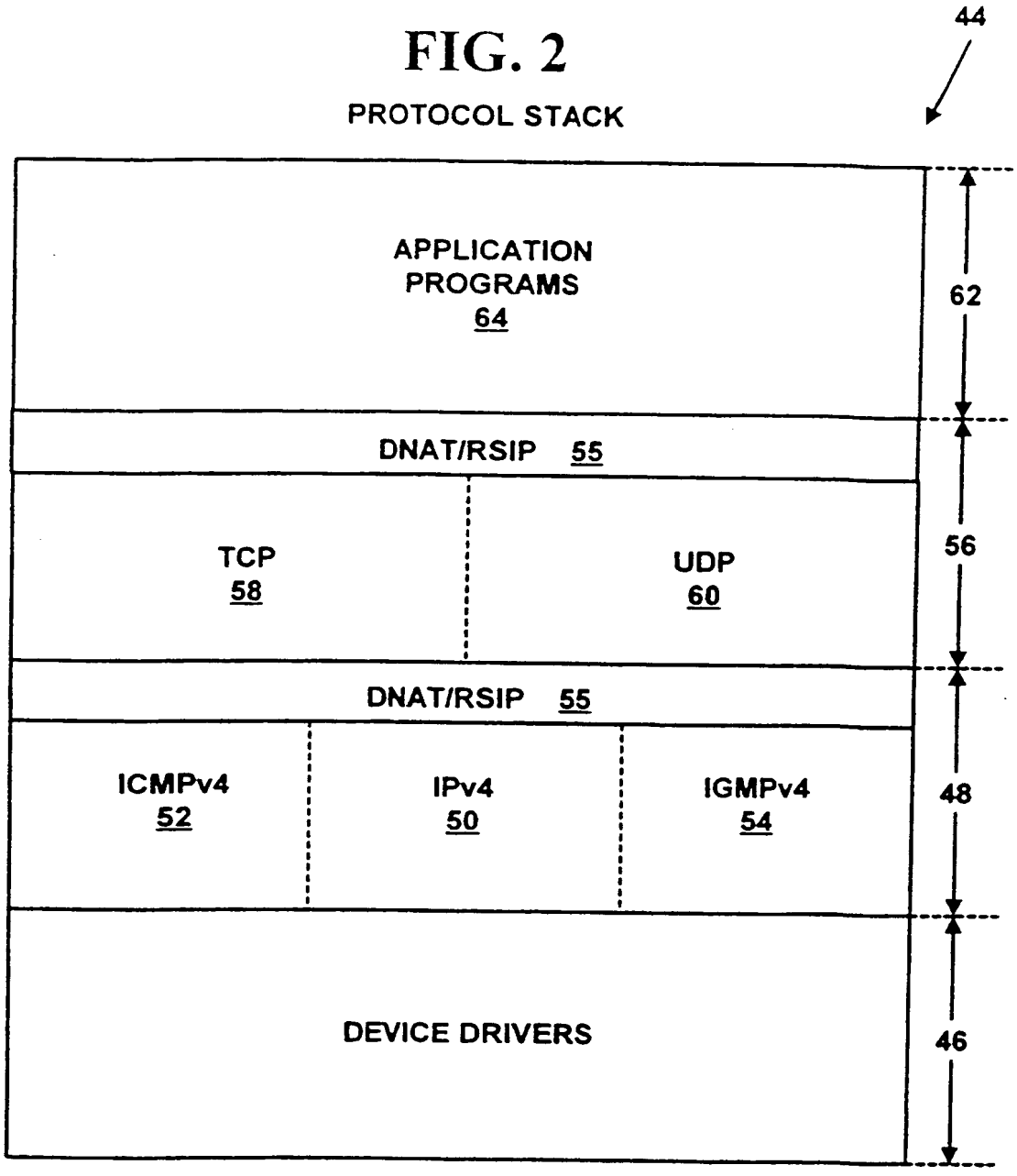

44
PROTOCOL STACK
APPLICATION PROGRAMS 64
62
DNAT/RSIP 55
TCP 58
UDP 60
56
DNAT/RSIP 55
ICMPv4 52
IPv4 50
IGMPv4 54
48
DEVICE DRIVERS
46

FIG. 3

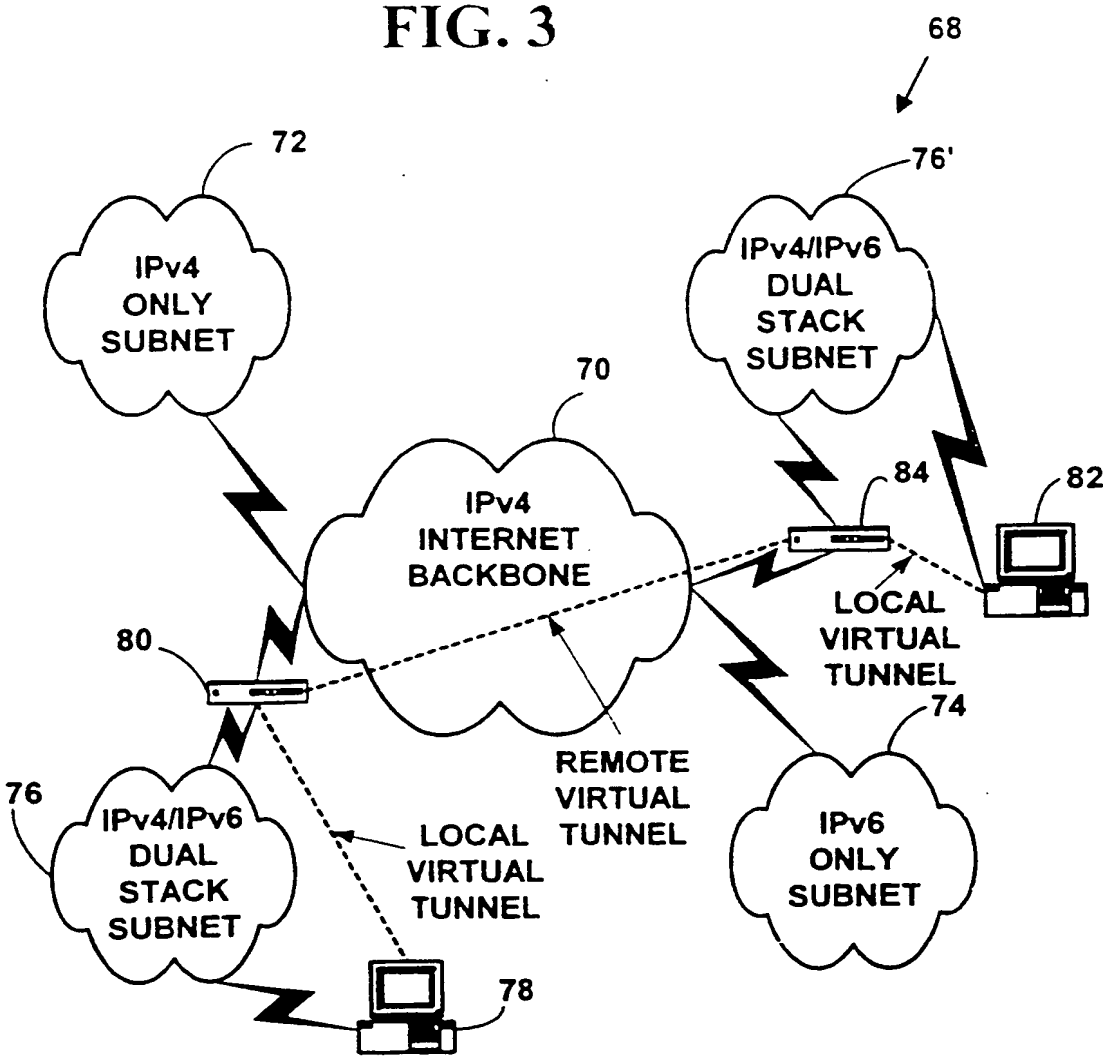

68
72
IPv4
ONLY
SUBNET
76'
IPv4/IPv6
DUAL
STACK
SUBNET
70
IPv4
INTERNET
BACKBONE
84
82
LOCAL
VIRTUAL
TUNNEL
80
74
REMOTE
VIRTUAL
TUNNEL
76
IPv4/IPv6
DUAL
STACK
SUBNET
LOCAL
VIRTUAL
TUNNEL
IPv6
ONLY
SUBNET
78

FIG. 4

DUAL PROTOCOL STACK

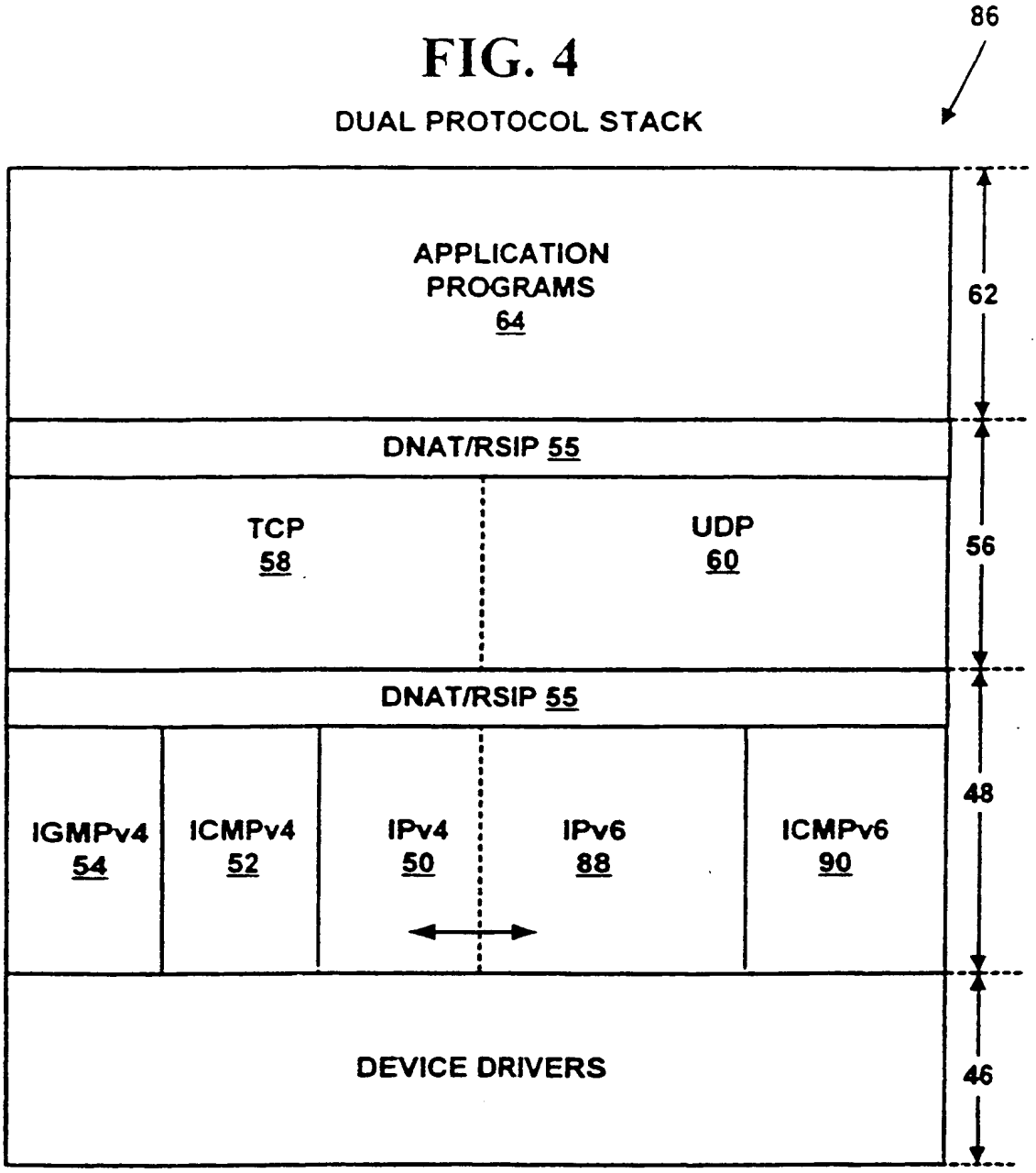


FIG. 5A

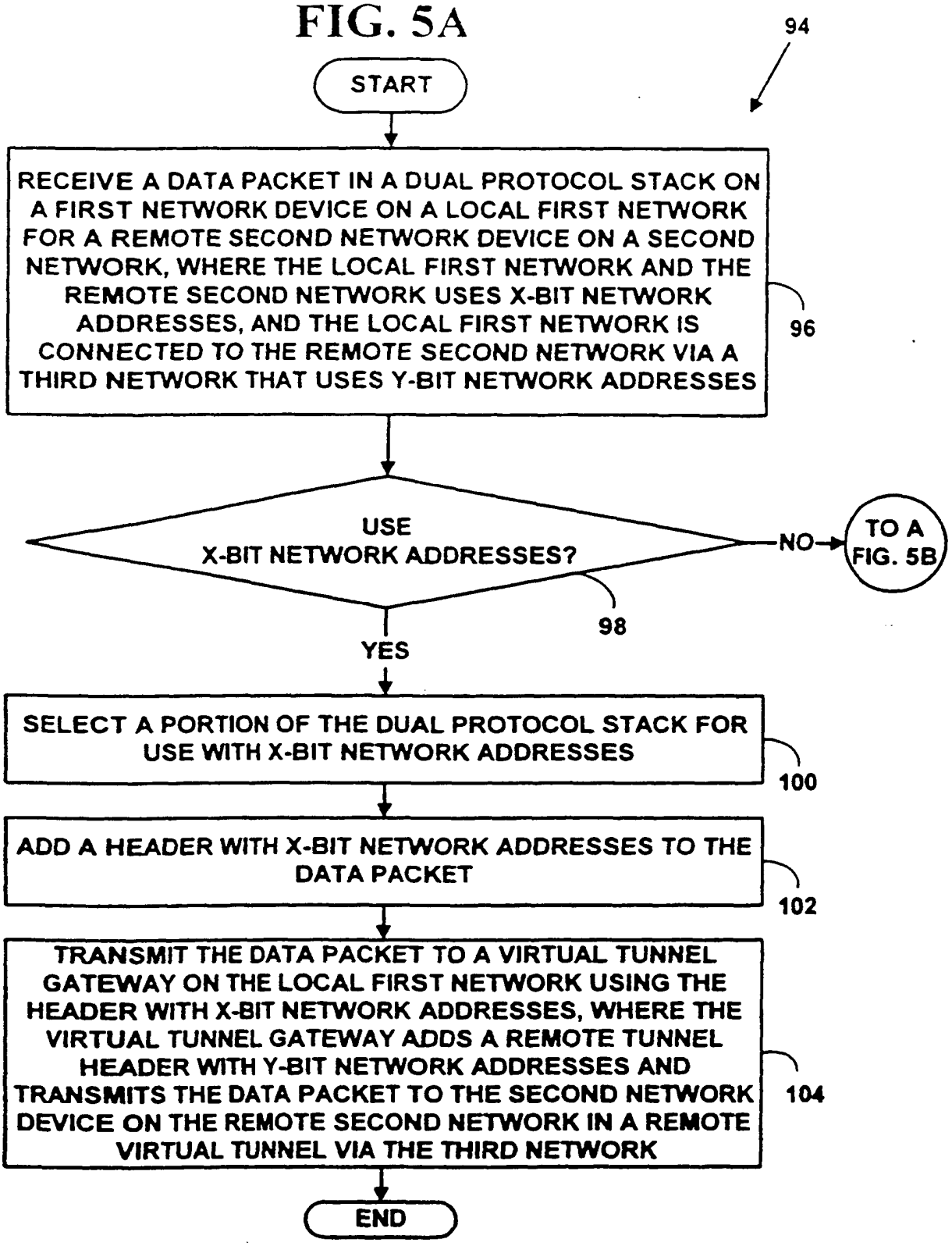


FIG. 5B

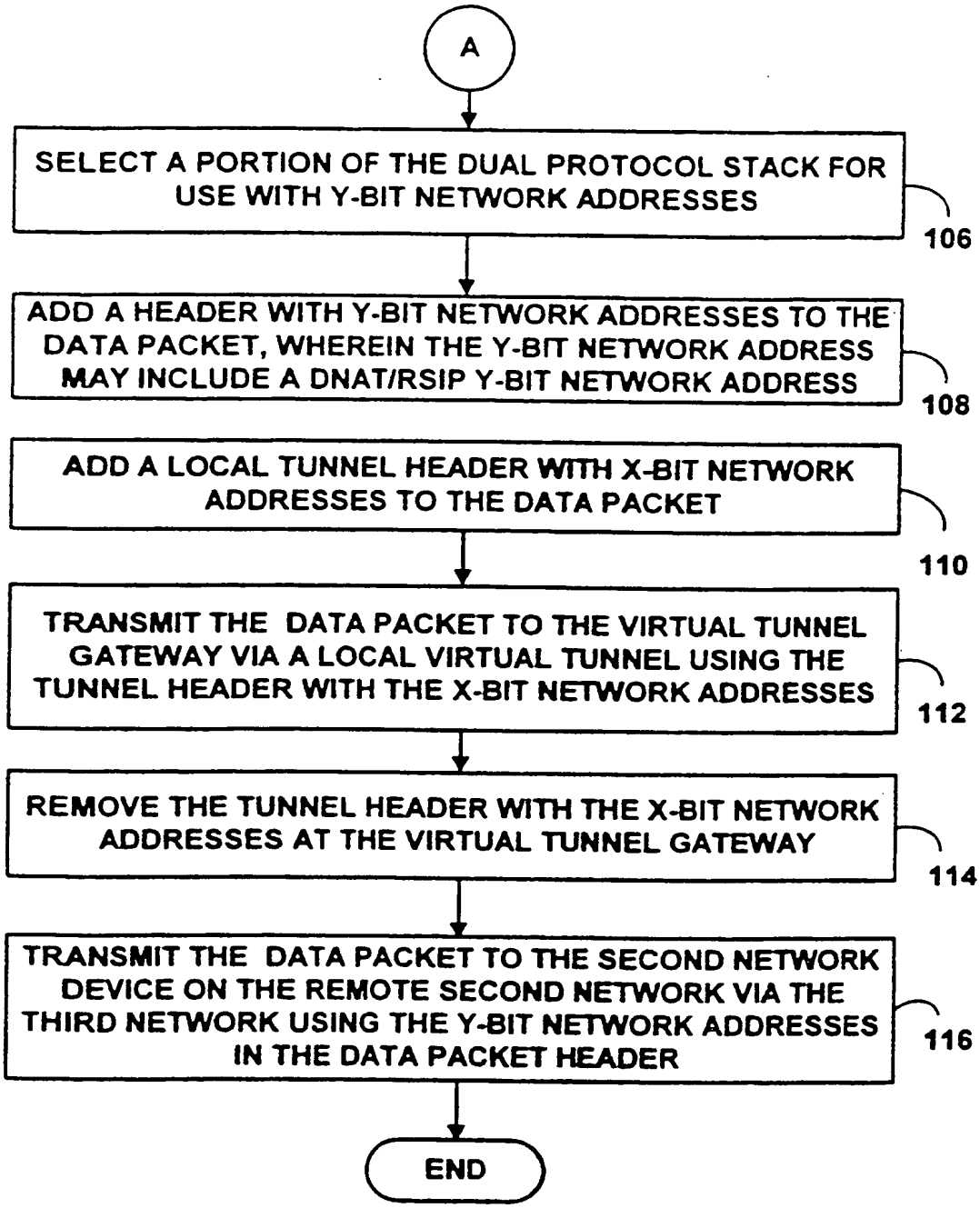


FIG. 6

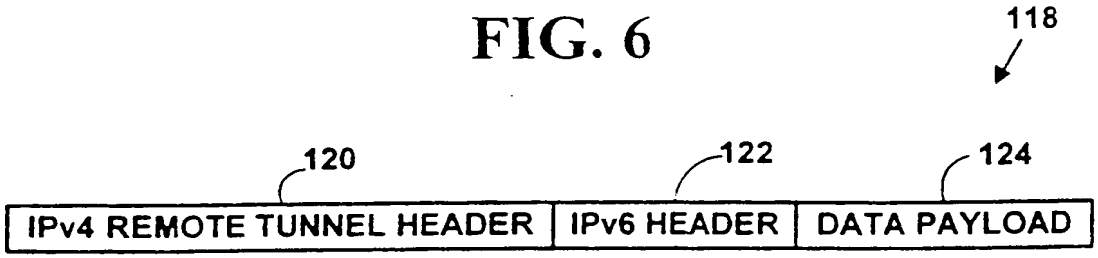
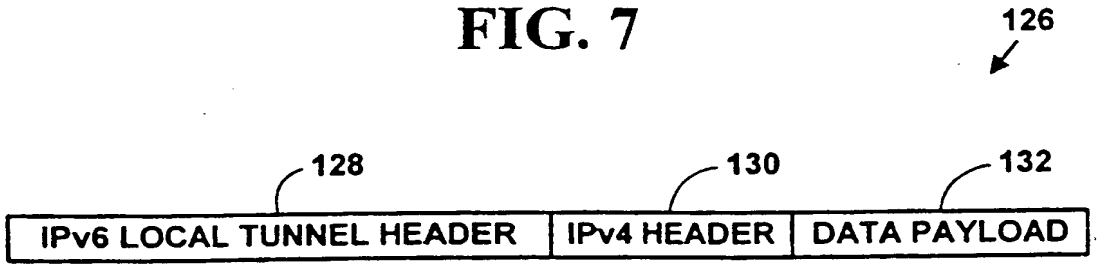


FIG. 7



INTERNATIONAL SEARCH REPORT

Int tional Application No
PCT/US 00/41211

A. CLASSIFICATION OF SUBJECT MATTER
IPC 7 H04L29/06

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
IPC 7 H04L

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practical, search terms used)
INSPEC, WPI Data, EPO-Internal, COMPENDEX, IBM-TDB

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category °	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	AFIFI H ET AL: "Methods for IPv4-IPv6 transition" PROCEEDINGS IEEE INTERNATIONAL SYMPOSIUM ON COMPUTERS AND COMMUNICATIONS (CAT. NO.PR00250), PROCEEDINGS IEEE INTERNATIONAL SYMPOSIUM ON COMPUTERS AND COMMUNICATIONS, RED SEA, EGYPT, 6-8 JULY 1999, pages 478-484, XP002159749 1999, Los Alamitos, CA, USA, IEEE Comput. Soc, USA ISBN: 0-7695-0250-4	1,3-7, 12-15, 19-22
Y	paragraphs '0002!,'02.1!,'0003!,'03.1!,'03.3!,'03.4!; figures 1,2	8,9,16, 17,23,24

-/--

[X] Further documents are listed in the continuation of box C.

[ ] Patent family members are listed in annex.

° Special categories of cited documents :

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier document but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art.

"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
8 February 2001	23/02/2001

Name and mailing address of the ISA	Authorized officer
European Patent Office, P.B. 5818 Patentlaan 2 NL - 2280 HV Rijswijk Tel. (+31-70) 340-2040, Tx. 31 651 epo nl, Fax: (+31-70) 340-3016	Huber, O

1

Form PCT/ISA/210 (second sheet) (July 1992)

INTERNATIONAL SEARCH REPORT

Int. tional Application No
PCT/US 00/41211

C.(Continuation) DOCUMENTS CONSIDERED TO BE RELEVANT

Category °	Citation of document, with indication,where appropriate, of the relevant passages	Relevant to claim No.
A	GILLIAN R.,NORDMARK E.: "Transition Mechanism for IPv6 Hosts and Routers" REQUEST FOR COMMENTS, 'Online! April 1996 (1996-04), pages 0-22, XP002159750 Internet Engeneering Task Force Retrieved from the Internet: <URL:http://www.ietf.org/rfc/rfc1933.txt?number=1933> 'retrieved on 2001-02-07! paragraphs '0003!,'0004!,'04.1!,'4.1.5!,'04.3! ---	1,11,14, 19,20
Y	BORELLA M., GRABELSKY D., IKHLAQ S., BRIAN P.: "Distributed Network Address Translation" INTERNET DRAFT, 'Online! October 1998 (1998-10), pages 0-24, XP002159751 Retrieved from the Internet: <URL:http://cph.telstra.net/ietf/old-ids/draft-borella-aatn-dnat-01.txt> 'retrieved on 2001-02-07! cited in the application page 1 -page 6 ---	8,9,16, 17,23,24
Y	BORELLA M., GRABELSKY D.: "Realm Specific IP: Protocol Specification" INTERNET DRAFT, 'Online! August 1999 (1999-08), pages 1-27, XP002159752 Retrieved from the Internet: <URL:http://cph.telstra.net/ietf/old-ids/draft-ietf-nat-rsip-protocol-02.txt> 'retrieved on 2001-02-07! cited in the application page 1 -page 10 -----	8,9,16, 17,23,24

1

Form PCT/ISA/210 (continuation of second sheet) (July 1992)

(19) Europäisches Patentamt
European Patent Office
Office européen des brevets

(11) EP 1 122 925 A1

(12) EUROPEAN PATENT APPLICATION

(43) Date of publication:
08.08.2001 Bulletin 2001/32

(51) Int Cl.[7]: **H04L 29/06**

(21) Application number: **00307408.5**

(22) Date of filing: **29.08.2000**

(84) Designated Contracting States:
AT BE CH CY DE DK ES FI FR GB GR IE IT LI LU MC NL PT SE
Designated Extension States:
AL LT LV MK RO SI

(30) Priority: **02.02.2000 US 497002**

(71) Applicant: **LUCENT TECHNOLOGIES INC.**
Murray Hill, New Jersey 07974-0636 (US)

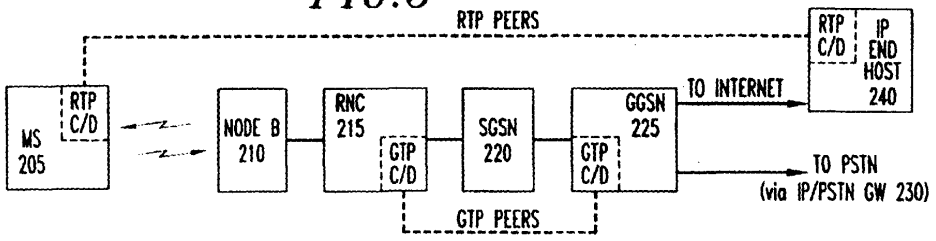
(72) Inventor: **Choo Chuah, Mooi**
Monmouth Country, New Jersey 07746 (US)

(74) Representative:
Watts, Christopher Malcolm Kelway, Dr. et al
Lucent Technologies (UK) Ltd,
5 Mornington Road
Woodford Green Essex, IG8 0TU (GB)

(54) Header compression for general packet radio service tunneling protocol (GTP)

(57) A UMTS (Universal Mobile Telecommunications System) core network supports a compression framework that provides for header compression of General Packet Radio Service Tunneling Protocol (GTP)-Encapsulated Packets. In particular, the GTP/UDP(User Datagram Protocol)/IP(Internet Protocol) header is compressed. In addition, the UMTS core network also supports RTP(Real Time Protocol)/UDP/IP header compression independent of the GTP/UDP/IP header compression.

FIG.5



EP 1 122 925 A1

Description

FIELD OF THE INVENTION

[0001] This invention relates generally to communications and, more particularly, to packet communications systems.

BACKGROUND OF THE INVENTION

[0002] As wireless systems continue to evolve, communications between a mobile switching center (MSC) and its base stations are moving to an Internet Protocol (IP) based transport mechanism. (As used herein, the term wireless systems refers to e.g., CDMA (code division multiple access), GSM (Global System for Mobile Communications), the proposed UMTS (Universal Mobile Telecommunications System), etc.) Given the nature of wireless communications, e.g., real-time voice, any IP-based transport needs to utilize a protocol that accommodates real-time applications.

[0003] One such protocol is the Real Time Protocol (RTP) (e.g., see H. Schulzrinne, R. Frederick, V. Jacobson, "*RTP: A Transport Protocol for Real-Time Applications*," RFC 1889). RTP is attractive since it is an available Internet Engineering Task Force (IETF) protocol for handling real-time streams. RTP traffic is encapsulated in UDP (user datagram protocol), and IP packets.

[0004] Unfortunately, the use of RIP/UDP/IP generates a large overhead when voice-over-IP applications are run over wireless networks since the voice payload is usually small (e.g. 10 to 20 bytes) while the RTP/UDP/IP header is 40 bytes.

SUMMARY OF THE INVENTION

[0005] Besides the large overhead associated with RTP/UDP/IP headers, this situation is further aggravated by the use of General Packet Radio Service Tunneling Protocol (GTP)-Encapsulated Packets. In this case, the GTP/UDP/IP overhead is about 980% with a voice payload of 10 bytes. Therefore, and in accordance with the invention, the GTP/UDP/IP header is compressed for transmission.

[0006] In an embodiment of the invention, a UMTS core network supports a compression framework that provides for compression of a GTP/UDP/IP header (referred to herein as "GTP header compression" or a "compressed GTP header"). In addition, the UMTS core network also supports RTP/UDP/IP header compression (referred to herein as "RTP header compression" or a "compressed RTP header") independent of the GTP header compression. As a result, the UMTS core network is able to more efficiently transport small multimedia RTP packets.

BRIEF DESCRIPTION OF THE DRAWING

[0007]

FIG. 1 shows a prior art uncompressed GTP encapsulated RTP packet;
FIG. 2 shows a UMTS network embodying the principles of the invention;
FIGs. 3 and 4 show illustrative protocol stacks for use in a mobile station;
FIG. 5 shows illustrative compressor/decompressor locations in the UMTS network of FIG. 2;
FIGs. 6 - 10 show illustrative message flows
FIGs. 11 - 13 show prior art IP, UDP and RTP header formats;
FIG. 14 shows an illustrative format for a compressed RTP header;
FIG. 15 shows an illustrative format for a compressed GTP header; and
FIG. 16 shows an illustrative high-level block diagram of a packet server for use in performing GTP header compression in accordance with the principles of the invention..

DETAILED DESCRIPTION

[0008] An illustrative format of an uncompressed GTP encapsulated RTP packet 10, as known in the art (release 97 version), is shown in FIG. 1. The GTP/UDP/IP header 11 comprises an IP/UDP header 12 and a GTP header 13. The GTP/UDP/IP header 11 sits on top of GTP payload 14 as known in the art. With a voice payload illustratively equal to 10 bytes (e.g., payload 15 of FIG. 1), this results in an overhead of about 980% as a result of the GTP/UDP/IP header 11. Therefore, and in accordance with the invention, the GTP/UDP/IP header is compressed for transmission. It can be observed from FIG. 1 that GTP payload 14 also transports an IP header and a UDP header. As known in the art, IP/UDP header 12 comprises origination and destination information with respect to the tunnel, while the IP header and UDP header of GTP payload 14 comprises origination and destination information with respect to the communi-

cations endpoints. (It should be noted that the TID value is based upon the release 97 version as defined in Global System for Mobile Communications (GSM) 04.08 document, Digital cellular telecommunications system (Phase 2+); mobile radio interface layer 3 - specification. The TID value includes 8 bytes: 12 bits MCC, 8 bits MNC, 40 bits MSIN and 4 bits NSAPI. The MCC, MNC, MSIN and NSAPI are parts of the IMSI defined in the GSM 04.08 document.)

[0009] An illustrative UMTS network 200 modified in accordance with the principles of the invention is shown in FIG. 2. Other than the inventive concept, the elements shown in FIG. 2 are well known and will not be described in detail. For example, UMTS network 200 comprises a radio access network (RAN), a core network (CN), a backbone network, and an illustrative endpoint represented by IP End Host 240. The backbone network comprises the Internet and the public switched telephone network (PSTN). The RAN comprises mobile station (MS) 205, node B 210 and radio network controller 215. (Although UMTS uses the term "node B," this is also referred to as a base station.) The CN comprises a serving GPRS support node (SGSN) 220, gateway GPRS support node (GGSN) 225 and element 230, which comprises a gatekeeper (GK) (a component in ITU H.323) and an IP/PSTN gateway (GW) (for translation between H.323 and the PSTN). Although shown as single block elements, the elements of UMTS network 200 include stored-program-control processors, memory, and appropriate interface cards (not shown). For the purposes of this description, an illustrative end-to-end connection is between MS 205 and IP End Host 240 (which are also referred to as endpoints). The term "packet server" as used herein refers to any packet processor, illustrations of which are the above-mentioned elements of UMTS 200.

[0010] In accordance with the invention, UMTS network 200 supports a compression framework that provides for GTP header compression. In addition, UMTS network 200 also supports RTP header compression independent of the GTP header compression. (As used herein the terms "GTP header compression" or "compressed GTP header" refer to compression of a GTP/UDP/IP header. Similarly, the terms "RTP header compression" or "compressed RTP header" refer to compression of an RTP/UDP/IP header.) Since GTP header compression is independent from RTP header compression, the GTP peers can be different from the RTP peers (but this is not required). This also provides some design flexibility since some multimedia traffic may not use RTP but purely UDP encapsulation. As a result, UMTS network 200 is able to more efficiently transport small multimedia packets. The description of the inventive concept continues below.

Overview of Header Compression

[0011] It should be understood that, in accordance with the invention, RTP header compression and GTP header compression can be negotiated independently between peers (described further below). Other than the inventive concept, compression/decompression techniques are well-known and will not be described herein. For example, typically a compressor/decompressor is a software module stored in a memory (not shown), e.g., of MS 205, and executed by a stored-program-controlled microprocessor (not shown), e.g., of MS 205. The software module uses conventional programming techniques, which, as such, will also not be described herein, to store shared information (described below) and format for transmission compressed, or reduced, forms of a GTP/UDP/IP header or an RTP/UDP/IP header (described below).

[0012] With respect to the mobile station, e.g. MS 205 of FIG. 2, two illustrative protocol stacks comprising a compressor/decompressor are shown in FIGs. 3 and 4. In the protocol stack of FIG. 3, the RTP compressor/decompressor is located between the IP layer and the RLC/MAC (radio link control/media access control) link layer. (For simplicity, the physical layer of the protocol stack is not shown.) In the protocol stack of FIG. 4, the RTP compressor/decompressor is located between the application layer and the RLC/MAC link layer. (Again, the physical layer is not shown.) (Although a form of RTP header compression is described below, it should be noted that the embodiment of FIG. 4 also represents that form of RTP header compression in which an RTP header is simply stripped off in its entirety (an RTP header is illustrated in FIG. 13).)

[0013] In a complementary fashion, a corresponding RTP compressor/decompressor and GTP compressor/decompressor are located in the UMTS network. An illustrative view of the location of the RTP compressor/decompressor and the GTP compressor/decompressor in UMTS network 200 is shown in FIG. 5. In FIG. 5, the RTP compressor/decompressor (C/D) is located in MS 205 and IP end host 240 (i.e., MS 205 and IP end host 240 are RTP peers), while the GTP compressor/decompressor (C/D) is located in RNC 215 and GGSN 225 (i.e., RNC 215 and GGSN 225 are GTP peers).

[0014] RTP is a point-to-point protocol. As such, for RTP header compression peers, it should be noted that this assumes that a link layer identifier (ID) is mapped to each RTP session identifier. One illustrative RTP session identifier comprises the IP destination address and the IP destination port number (these are of the endpoints), the SSRC identifier (e.g., see FIG. 13), and the UDP destination port (e.g., see FIG. 12). If ATM (asynchronous transfer mode) is used as transport, an illustrative link layer ID is the associated VPI/VCI (Virtual Path Identifier/Virtual Connection Identifier). The mapping of the link layer ID and the RTP session occurs within each RTP peer.

[0015] It should also be noted that an RTP compressor/decompressor can alternatively be put in the radio access

network. e.g., in RNC 215, or even in the core network, e.g., at SGSN 220.

[0016] Within the core network, it is preferable (though not required) to put the GTP compressor/decompressor in GGSN 225. If the GTP compressor/decompressor is located in SGSN 220, handovers (also known in the art as "hand-offs") due to SRNS (Serving Radio Network Subsystem) relocation may still have to be accounted for. (As known in UMTS, an SRNS includes not only a particular RAN but also supporting elements, e.g., a data base (not shown).) However, with the GTP compressor/decompressor located at GGSN 225, no context transfer is required even for the case of SRNS relocation.

[0017] Turning now to FIGs. 6 - 10, illustrative message flows for using GTP header compression and RTP header compression in UMTS network 200 are shown. Other than the inventive concept, the description that follows utilizes well-known UMTS message flows, which are not described herein. In FIGs. 6 - 10 it is assumed that the GTP header peers are RNC 215 and GGSN 225, while the RTP header peers are MS 205 and IP End Host 240.

[0018] FIG. 6 illustrates how a mobile station, e.g., MS 205 of FIGs. 2 and 5, negotiates GTP header compression/decompression. After the "Attach Procedures" are performed between MS 205 and RNC 215 (as known in the art), MS 205 transmits to SGSN 220 an "Activate PDP (packet data protocol) context" request message modified, in accordance with the invention, to include a GTP header compression request represented by a predefined identifier designated as "GTP_Comp." In response, SGSN 220 sends a "Create PDP context" request message (modified to include the "GTP_Comp" identifier) to GGSN 225 to signify a request for GTP header compression. GGSN 225 responds with a "Create PDP context" response message as an acknowledgment (modified to include the "GTP_Comp" identifier). Upon receipt of the "Create PDP context" response message, SGSN 220 sends an "Activate PDP context" response message (modified to include the "GTP_Comp" identifier) to MS 205.

[0019] As noted above, in order to establish a GTP header compression context, either a GTP_Compressed flag (e.g., a predefined bit pattern) or a GTP Header Compression Context Information Element (IE) is added to the existing message set. This GTP Header Compression_Context IE will comprise the GTP full header information. If this element is present, one need not send a full GTP header to establish the GTP header context. Otherwise, one may need to send one or more packets with full GTP header to establish GTP header context. (It should be noted that for the case where the GTP compressor/decompressor is located at an SGSN, and an SRNS relocation occurs resulting in a change of SGSN, the new SGSN can send a GTP context enquiry message to the old SGSN and the old SGSN can reply with the appropriate GTP context response message so that the new SGSN can now be the new compressor/decompressor point for GTP header compression.)

[0020] Turning now to FIG. 7, illustrative packet flows are shown. For GTP header compression, at least one packet with a full GTP header is sent to establish the GTP compressed header context (FIG. 7, (A)) between the GTP peers (here, represented by RNC 215 and GGSN 225). As noted, packets are communicated using GTP between GGSN 225 and RNC 215 (i.e., GTP terminates at GGSN 225 and RNC 215). Once the GTP header compression is negotiated, each GTP peer, e.g., GGSN 225 formats data packets in accordance with GTP and then compresses the GTP header before transmitting the packets (described below) to its GTP peer, here RNC 215, which uncompresses the GTP header and recovers the payload (which may or may not include RTP) for transmission to MS 205. Similarly, in the other direction, RNC 215 compresses the GTP header for transmission to GGSN 225, which uncompresses the GTP header and recovers the payload. Packets are communicated between GGSN 225 and IP End Host 240 with a link layer header as known in the art. Subsequent to GTP header compression negotiation, RTP header compression may optionally be negotiated between MS 205 and IP End Host 240 (FIG. 7, (B)) in a similar fashion as that shown for the GTP header compression negotiation of FIG. 6.

[0021] Illustrative RTP header compression context message exchanges are shown in FIG. 8. (Again, modification of existing signaling messages are assumed. As such predefined bit values are added to existing message sets to identify the additional message requirements, e.g., that this is an "RTP context set up" message.) Initially, two RTP peers exchange signaling messages to set up the RTP header compression context. An "RTP context set up" request message is sent from one RTP peer, e.g., MS 205, to the other RTP peer, e.g., IP End Host 240. An "RTP context Set up" response message completes the handshake. Whenever there is a change in the RTP context, the appropriate context update code is used in the first byte of the compressed RTP header (described below) to indicate the additional changed (or delta) information carried within the RTP compressed header. Thus, an "RTP context update" request message is an implicit message in the RTP compressed header. However, an "RTP context update" response message is optional. (It should be noted that the RTP header compression between the two RTP peers (here represented by MS 205 and IP End Host 240) can exchange out-of-band signaling messages to turn on the RTP header compression. Again, since the GTP header compression and the RTP header compression are independent of each other, it is not necessary for RTP header compression to be negotiated (in which case, the packets comprise a compressed GTP header and an uncompressed RTP payload.) Similarly, it is not necessary for GTP header compression to be negotiated notwithstanding the use of RTP header compression.

[0022] Returning to FIG. 7, once the RTP header compression context is set up, subsequent packets are sent using GTP header compression and RTP header compression (FIG. 7, (C)) (as noted, assuming both GTP header compres-

sion and RTP header compression are turned on). In the event RTP context re-synchronization is required, the receiving RTP peer can send a "RTP context repair" message to the sending RTP peer. This is illustrated in FIG. 9, (D), where the receiving RTP peer is represented by IP End Host 240 and the sending RTP peer is represented by MS 205. The sending RTP peer then sends one or more RTP packets with full header (FIG. 9 (E)) followed by compressed RTP packets (FIG. 9 (F)).

[0023] Similarly, when GTP packets are lost, the receiving GTP peer can send a "GTP context repair" message to the sending GTP peer. This is illustrated in FIG. 10, (G), where the receiving GTP peer is represented by GGSN 225 and the sending GTP peer is represented by RNC 215. The sending GTP peer then sends one or more packets with a full GTP header (FIG. 10, (H)) followed by packets with compressed GTP headers (FIG. 10, (I)). (It should be observed that in this example, RTP header compression synchronization was not lost.)

[0024] The above-described context repair mechanism for either GTP header compression or RTP header compression is performed whenever there are missing packets. It should be noted that one can set a predefined time interval threshold beyond which an explicit context repair message is sent to the sender to re-synchronize.

[0025] Whenever either GTP peer wishes to tear down the GTP context, they can send a GTP context tear down message (not shown). Similarly, the RTP peers can tear down the RTP context via the transmission of an RTP context tear down message (not shown).

RTP Header Compression

[0026] Although not directly applicable to a UMTS environment, there are existing proposals to reduce the 40-byte RTP/UDP/IP header to 4 - 5 bytes (e.g., see S. Casner and V. Jacobson, "Compressing IP/UDP/RTP Headers for Low-Speed Serial Links," IETF RFC2508; and L. Jonsson and M. Degermark and H. Hannu and K. Svanbro, "Robust Checksum-based Header Compression", IETF internet draft, Oct 1999). As such, while FIGs. 11 - 13 show prior art IP, RTP and UDP header formats for reference purposes, the details of such are not described herein.

[0027] Generally, with respect to an IP header (assuming use of IPv4, the IP version in use today), only the total length, packet ID (identification) and header checksum fields will normally change. However, the total length is redundant since the length is also provided by the link layer. The packet ID usually increments by one or a small number for each packet. If it was assumed that there was no IP packet fragmentation, this also would not need to be communicated. However, in order to maintain lossless compression, changes in the packet ID may be transmitted.

[0028] With respect to a UDP header, the length field is redundant since the IP total length field and the length are indicated by the link layer. The UDP check-sum field will be a constant zero if the source elects not to generate UDP checksums. Otherwise, I have observed that the UDP checksum must be communicated intact in order to preserve the lossless compression.

[0029] With respect to an RTP header, the SSRC (synchronization source) identifier is constant in a given context since that is part of what identifies the particular context. For most packets, only the sequence number and the timestamp will change from packet to packet. If packets are not lost or misordered upstream from the compressor, the sequence number will increment by one for each packet. For audio packets of constant duration, the timestamp will increment by the number of sample periods conveyed in each packet. For video, the timestamp will change on the first packet of each frame, but then stay constant for any additional packets in the frame. If each video frame occupies only one packet, but the video frames are generated at a constant rate, then again the change in the timestamp from frame to frame is constant. It should be noted that in each of these cases the second-order difference of the sequence number and timestamp fields is zero, so the next packet header can be constructed from the previous packet header by adding the first-order differences for these fields that are stored in the session context along with the previous uncompressed header. When the second-order difference is not zero, the magnitude of the change is usually much smaller than the full number of bits in the field, so the size can be reduced by encoding the new first-order difference and transmitting it rather than the absolute value.

[0030] The M bit is set on the first packet of an audio talkspurt and the last packet of a video frame. If it were treated as a constant field such that each change required sending the full RTP header, this would reduce the efficiency of header compression significantly. Therefore, as described further below, an RTP compressed header will carry the M bit explicitly.

[0031] If the packets are flowing through an RTP mixer, most commonly for audio, then the CSRC list and CC count will also change. However, the CSRC list will typically remain constant during a talkspurt or longer, so it need be sent only when it changes.

[0032] An illustrative format for a compressed RTP header is shown in FIG. 14 for use as part of the RTP header compression protocol. As noted above, it is assumed an RTP peer transmits the compressed RTP header and maintains a collection of shared information (e.g., stored in a memory (not shown) in a consistent state between the compressor and decompressor). The compressed RTP header comprises the following fields:

- a Context Update Code (one byte);
- an M field (one bit) for the RTP M bit;
- a time clicks field (seven bits);
- a UDP checksum field (two bytes);
- an IP packet ID (two bytes);
- a CSRC list (two bytes); and
- an RTP header extension (two bytes).

[0033] The value of the Context Update Code field indicates what information is included in the RTP compressed header as shown in FIG. 14. The minimal length for the RTP compressed header is 2 bytes. The RTP timestamps are replaced by a timeclick number (1 byte). If the UDP checksum, IPv4 Packet ID, CSRC list and RTP header extension need to be included, the compressed RTP header will be longer as shown in FIG. 14. However, most of the time, the compressed RTP header will only be 2 bytes (the context update code byte and the M + timeclick byte).

[0034] With respect to the shared information stored in each RTP peer, there is a separate session context for each IP/UDP/RTP packet stream, as defined by a particular combination of the IP source and destination addresses, UDP source and destination ports, and the RTP SSRC field (described earlier). The number of maintained session contexts may be negotiated between the compressor and decompressor. Once can map each RTP context to a GTP TID (GTP tunnel ID) (the maximum number that can be negotiated is 65536). Each RTP header compression context has its own separate sequence number space so that a single packet loss need only invalidate one context.

[0035] The shared information in each RTP header compression context comprises the following items:

- the full IP, UDP and RTP headers, possibly including a CSRC list, for the last packet sent by the compressor or reconstructed by the decompressor; and
- the first-order difference for the IPv4 ID field, initialized to 1 whenever an uncompressed IP header for this context is received and updated each time a delta IPv4 ID field is received in a compressed packet.

[0036] As mentioned above, and shown in FIG. 14, there is a timeclicks field in the compressed RTP header which replaces the RTP timestamps field of an RTP header (e.g., see FIG. 13). As such, a mechanism is required o compute, or recover, the timestamp value and sequence number in an RTP receiving peer from the timeclicks field value. As such, the following definitions are made:

sd: sample duration (in ms);
TS_{old}: timestamp number for the previous packet;
TS_{new}: timestamp number for this packet;
WT_{old}: wall clock reading for the previous packet (a wall clock is simply a network reference clock);
WT_{new}: wall clock reading for this packet;
TN_{old}: timeclick number of previous packet in compressed header;
TN_{new}: timeclick number of this packet in compressed header;
SN_{old}: RTP sequence number of previous packet;
SN_{new}: RTP sequence number of this packet;
T: time period represented using n bits (a cycle period), in units of sample duration, $T=2^n$ samples ($T = 2^n(sd)$ milliseconds (ms)); and
M: the value of M bit in compressed header.

[0037] The following equations are used to compute, or recover, the timestamp value and sequence number in an RTP receiving peer from the timeclicks field value:

$$TS_{new} = TS_{old} + 1; \quad \text{if } M = 0 \text{ and } \delta_{tick} = 1; \text{ and} \tag{1}$$

$$= TS_{old} + (\delta_{cycle} * T + \delta_{tick}) \quad \text{otherwise;} \tag{2}$$

$$SN_{new} = SN_{old} + \delta_{tick}, \text{ if } \delta_{tick} != 1, \text{ and } \delta_{tick} < T/4 \text{ and } M != 1; \tag{3}$$

where

$$\delta_{tick} = (T + TN_{new} - TN_{old}) \bmod T; \text{ (where modT is a modulo T operation);} \quad (4)$$

$$\delta_{wtick} = (WT_{new} - WT_{old}) \bmod T; \quad (5)$$

$$\delta'_{cycle} = \lfloor (WT_{new} - WT_{old})/T \rfloor; \text{ (where "}\lfloor\ \rfloor\text{" represent the "floor")} \quad (6)$$

$$\delta_{cycle} = \delta'_{cycle} - 1, \quad \text{if } \delta_{wtick} < \delta_{tick}, \text{ and } \delta_{tick} - \delta_{wtick} \geq T/2; \quad (7)$$

$$= \delta'_{cycle} + 1, \quad \text{if } \delta_{wtick} < \delta_{tick}, \text{ and } \delta_{wtick} - \delta_{tick} \geq T/2; \text{ and} \quad (8)$$

$$= \delta'_{cycle}, \quad \text{otherwise.} \quad (9)$$

[0038] During a talkspurt, the timeclicks field value will increase by one sample. During a silent period, the timeclicks field will increase by the idle period (expressed in terms of the number of samples).

[0039] For the timestamp field, the major problem to solve is what to do for the case when the 7-bit timeclick number wraps around. During a silent period, if there is no packet sent by the compressor, the time elapse for the silent period must be detected (in terms of how many clock cycles has passed). Illustratively, a wall clock as known in the art is used to overcome this problem (as shown above, e.g., WT_{old} and WT_{new}). This separate wall clock at the decompressor (or receiving RTP peer) is used to count the cycles. This wall-clock runs at a coarse granularity, e.g. it only increases by 1 for every $T/4$ period where T is the time period of a cycle represented using 7 bits.

[0040] With respect to the RTP sequence number field, if sequencing is required, a compressed header with sequence number should be included every $T/4$ samples and at the beginning of every talkspurt. A context repair message can be sent to request a full RTP header if necessary. If the timeclick value exceeds T/4, and there are lost packets, the RTP sequence number can't be updated appropriately until the RTP receiver gets a packet with sequence number information.

[0041] An illustration of how this update algorithm works to provide timestamp and sequence number recovery is shown below. It is assumed that 18 packets are transmitted from, e.g., MS 205, with consecutive sequence numbers 1 to 18. At the RTP receiver, e.g., IP End Host 240, only packet 18 is received (i.e., packets 7 to 17 are lost).

[0042] In the first example, its is assumed that the wall clock value is 3 (meaning $3(T/4)$) when packet with sequence number 6 is received and that $TS_{old} = 110$. When packet 18 is received, the wall clock value is 6. The timeclicks value at receipt of packets 6 and 18 are 110 and 75, respectively. Using the equations above, the following calculations result,

$$TS_{old} = 110;$$

$$\delta_{tick} = (128 + 75 - 110) \bmod T = 93;$$

$$\delta_{wtick} = (6(T/4) - 3(T/4)) \bmod T = 3(T/4) = 96;$$

$$\delta'_{cycle} = \lfloor (6(T/4) - 3(T/4))/T \rfloor = 0;$$

$$\delta_{cycle} = 0;$$

and

$$TS_{new} = 110 + 93 = 203$$

[0043] In the following second example, it is assumed that $TS_{old} = 38$ and that the wall clock value for packet 6 is 5, while the wall clock value for packet 18 is 9. The timeclicks value at receipt of packets 6 and 18 are 38 and 32, respectively.

$$TS_{old} = 38;$$

$$\delta_{tick} = (128 + 32 - 38) \bmod T = 122;$$

$$\delta_{wtick} = (9(T/4) - 5(T/4)) \bmod T = 0;$$

$$\delta'_{cycle} = \lfloor (9(T/4) - 5(T/4))/T \rfloor = 1;$$

$$\delta_{cycle} = 0;$$

and

$$TS_{new} = 38 + 122 = 160.$$

[0044] In the following third example, it is assumed that $TS_{old} = 57$ and that the wall clock value for packet 6 is 6, while the wall clock value for packet 18 is 9. The timeclicks value at receipt of packets 6 and 18 are 57 and 28, respectively.

$$TS_{old} = 57;$$

$$\delta_{tick} = (128 + 28 - 57) \bmod T = 29;$$

$$\delta_{wtick} = (9(T/4) - 6(T/4)) \bmod T = 3(T/4) = 96;$$

$$\delta'_{cycle} = \lfloor (9(T/4) - 6(T/4))/T \rfloor = 0;$$

$$\delta_{cycle} = 1;$$

and

$$TS_{new} = 57 + 122 = 214.$$

GTP Header Compression

[0045] FIG. 15 shows an illustrative format for a compressed GTP header. The compressed GTP header comprises a version field (Vers, 3 bits), a payload type field (PT, 1 bit), an extension bit field (E, 1 bit), a sequence number field (S, 1 bit), a tunnel identifier (TID) present field (T, 1 bit), an SNDCP present field (N, 1 bit), a message type field (Msg Type, 1 byte), a two byte length field, a two to four byte tunnel identifier (TID) field, a two byte sequence number field, a four bit extension code filed, a four bit extension length field, and an extension content field.

[0046] With respect to the TID field, the most significant bit is used to indicate the size of the TID field. If the value of the most significant bit is 0, then the TID field size is 2-bytes. If the value of this bit is one, then the TID field size is four bytes. In addition, the TID field is present only if the value of the T bit field is SET, i.e., equal to a binary one. The

E-bit field is used to indicate if an extension header exists. Each extension header comprises a 4 bit extension header type (Ext. Type) field, a 4 bit extension header length (Ext. length) field, and an extension content (Ext. Content) field (the size of which is indicated by the value of the extension header length field). The value of the extension header length field is expressed in terms of multiples of 2 bytes. For example, if a UDP checksum needs to be present, an appropriate extension header type is defined for this and the extension header length is 1 (meaning the size of the extension content field is 2 bytes since a UDP checksum is 2 bytes long).

[0047] Since the extension header type field is four bits wide, 16 types of extensions can be defined. Since the extension header length field is four bits wide, the extension content field has a maximum size of 32 bytes (i.e., 16 two byte multiples). It should be noted that if more extension types need to be allocated, the size of these fields can be adjusted, e.g., to a size of one byte each.

[0048] The shared information in each GTP header compression context (i.e., stored at each GTP peer) comprises the following items (these items are initialized based on the values received in the initial "full" GTP header):

- the full IP and UDP headers (also, should a UDP checksum be required, the receiving GTP peer simply re-calculates it based upon the received UDP data);
- the two byte flow label and the LLC Frame number; and
- the TID value.

[0049] As noted above, the GTP header compression format shown in FIG. 15 supports either two or four byte optional TID values. (As noted above, the 97 release version of a TID is 8 bytes. However, TID values may be redefined in the future to comprise fewer bytes (hence the option of having either TID values of two or four bytes in the compressed GTP header). The TID field in the GTP header compression format of FIG. 15 is optionally used (via the value of the T bit) for updating TID information.

[0050] Turning briefly to FIG. 16, a high-level block diagram of a representative packet server for use in accordance with the principles of the invention is shown. Packet server 605 is a stored-program-control based processor architecture and includes processor 650, memory 660 (for storing program instructions and data, e.g., for communicating in accordance with the above-mentioned GTP header compression, etc.) and communications interface(s) 665 for coupling to one or more packet communication facilities as represented by path 666.

[0051] The foregoing merely illustrates the principles of the invention and it will thus be appreciated that those skilled in the art will be able to devise numerous alternative arrangements which, although not explicitly described herein, embody the principles of the invention and are within its scope. For example, although illustrated in the context of UMTS, the inventive concept is applicable to any wireless system (e.g., UMTS, etc.) or application that requires use of a tunneling protocol.

Claims

1. A method for use in a packet server, the method comprising the steps of:

   negotiating compression of a GTP/UDP/IP header of a General Packet Radio Service based Tunneling Protocol (GTP) with a GTP peer, the GTP/UDP/IP header comprising GTP, User Datagram Protocol (UDP), and Internet Protocol (IP) header information;
   initially transmitting the GTP/UDP/IP header to the GTP peer; and
   subsequently transmitting a compressed version of the GTP/UDP/IP header to the GTP peer.

2. The method of claim 1 wherein the compressed GTP/UDP/IP header comprises at least a tunnel identifier field that is less than or equal to four bytes in length.

3. The method of claim 1 wherein the compressed GTP/UDP/IP header comprises at least, an extension bit field, which indicates whether an extension field is present or not, a tunnel identifier present field, which indicates whether a tunnel identifier field is present or not.

4. The method of claim 1 further comprising the steps of:

   formatting data packets in accordance a Real Time Protocol (RTP) also having UDP and IP header information; and
   compressing the RTP/UDP/IP header before transmitting the packets wherein one field of the compressed RTP/UDP/IP header defines whether a UDP checksum field is present or not.

FIG. 1
(PRIOR ART)

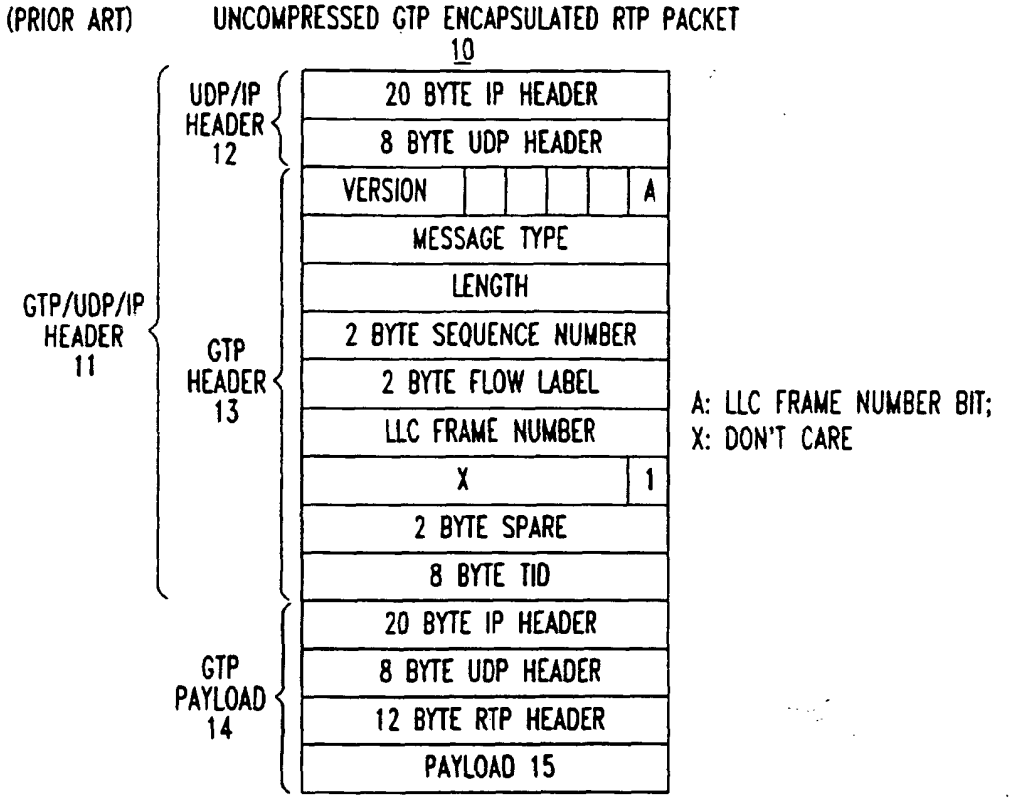
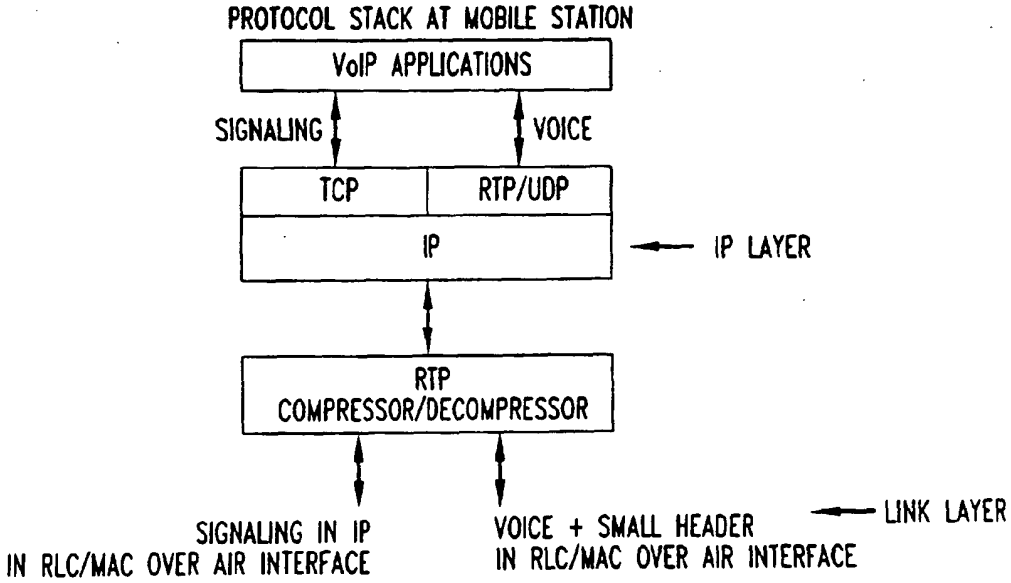


FIG. 3



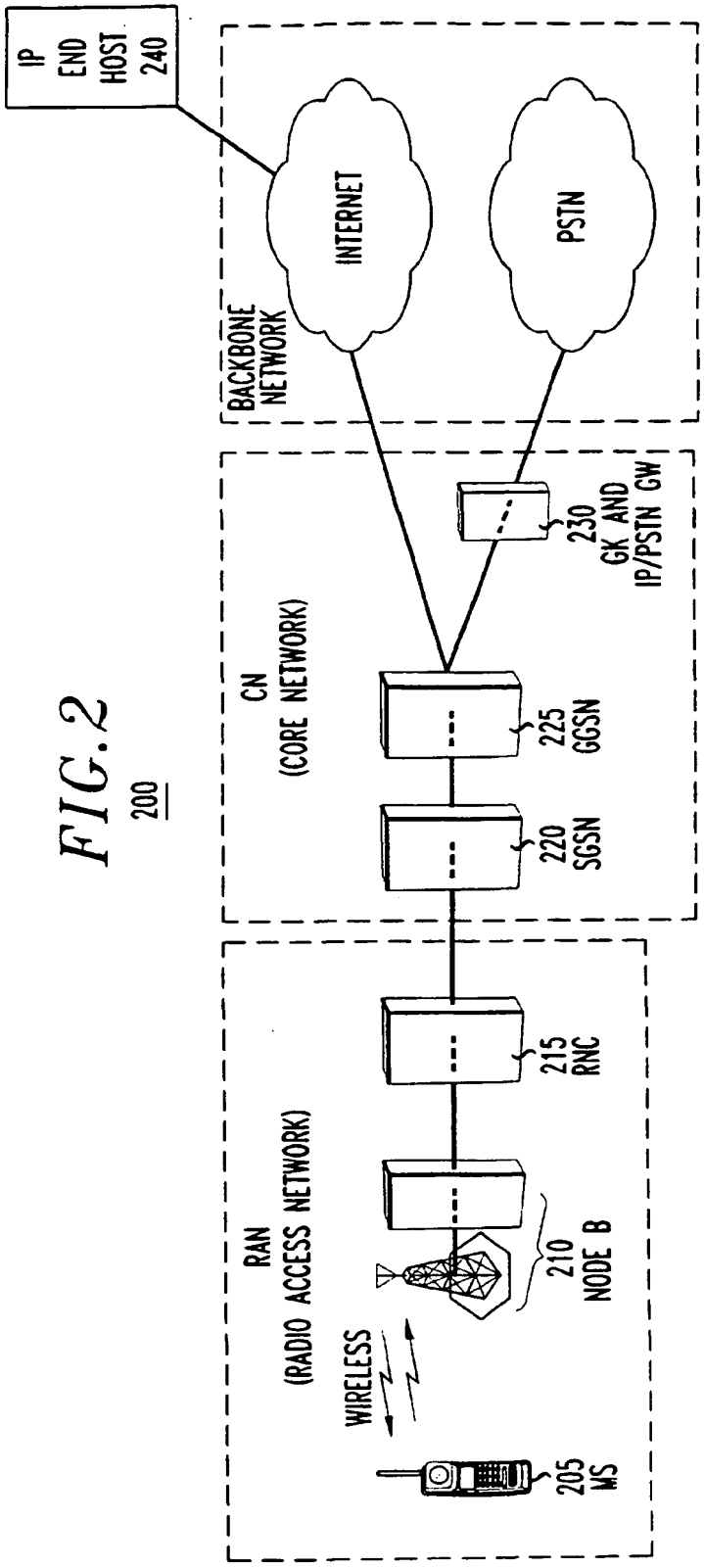


FIG.2

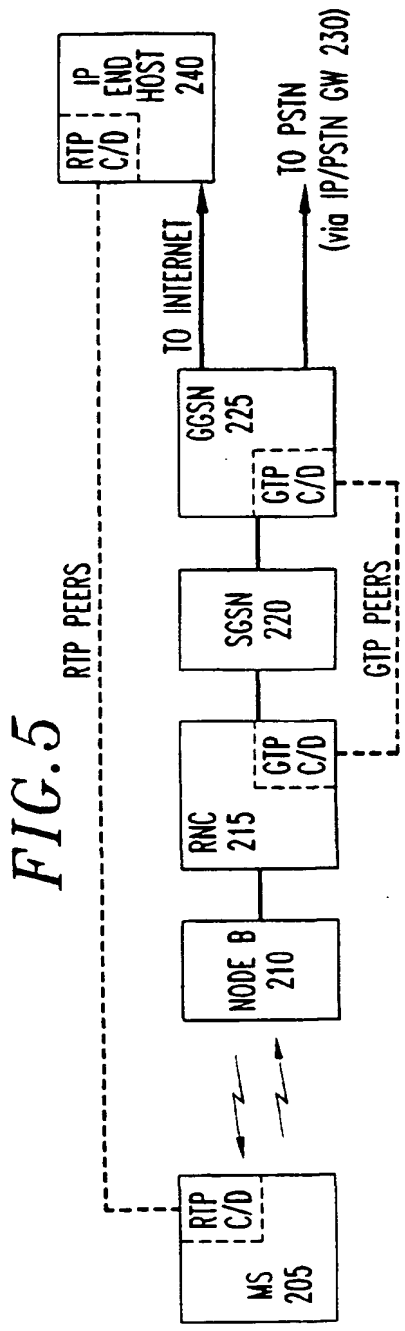


FIG.5

FIG. 4

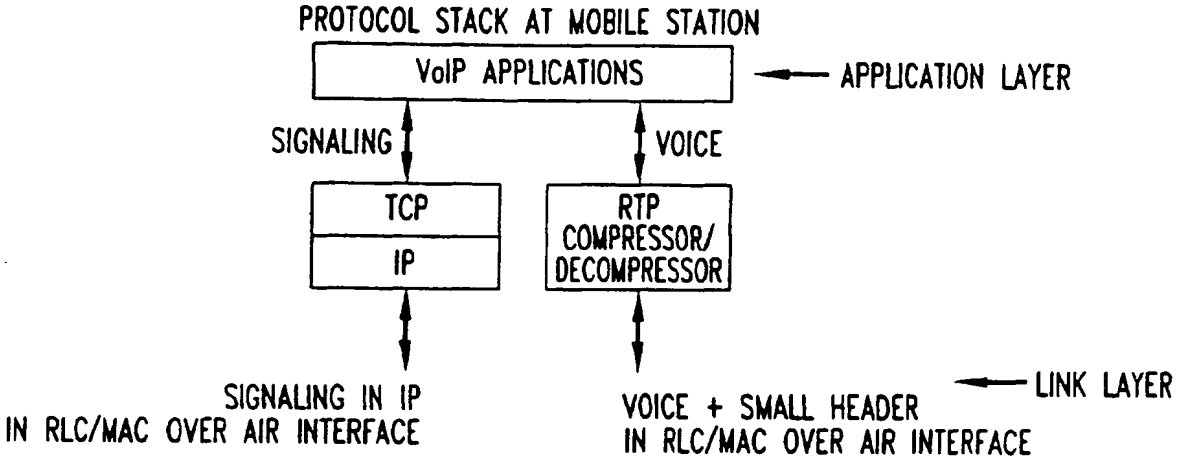


FIG. 6

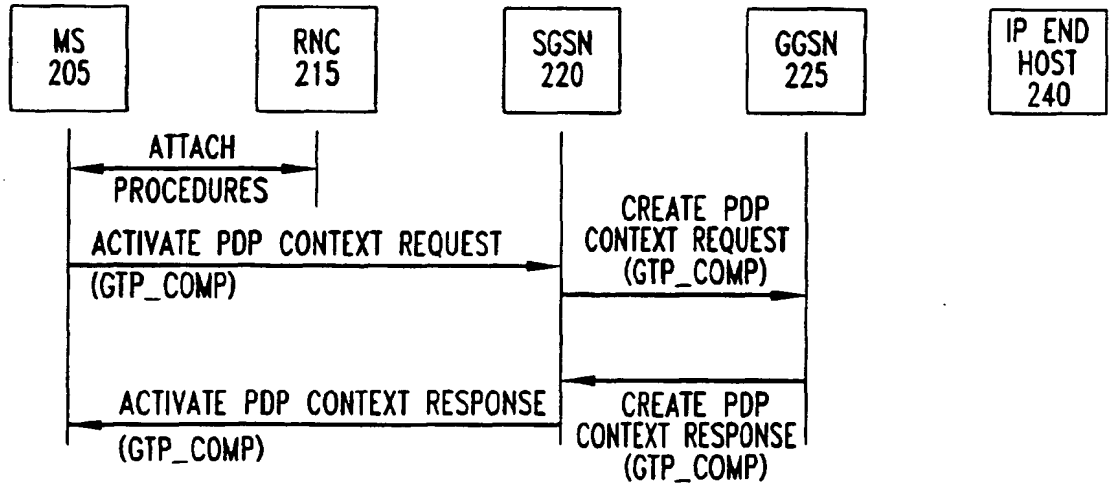


FIG. 7

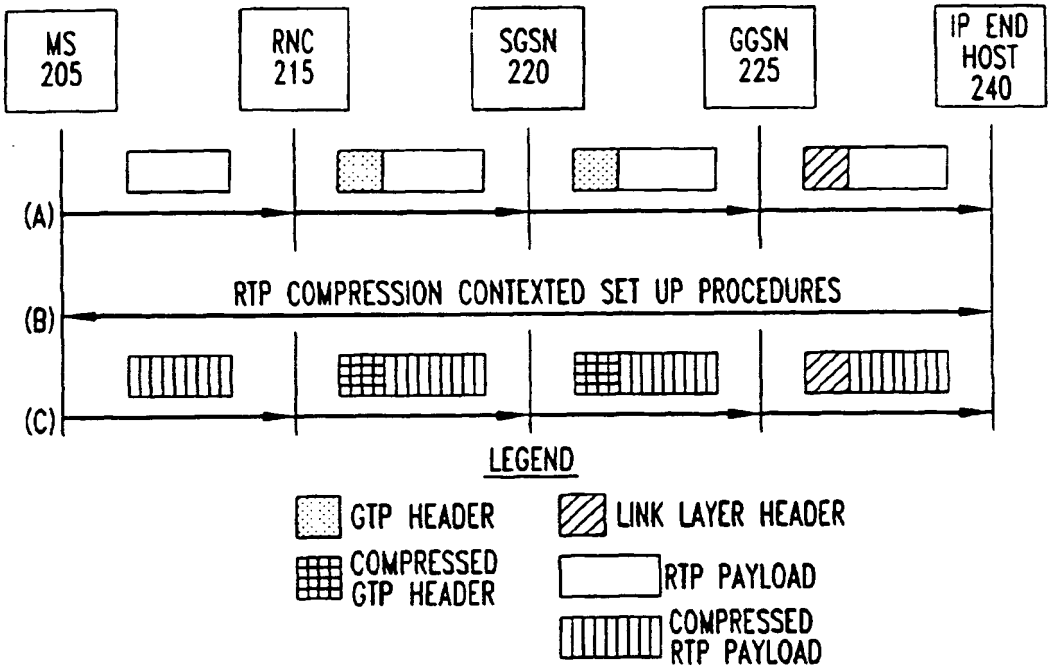


FIG. 8

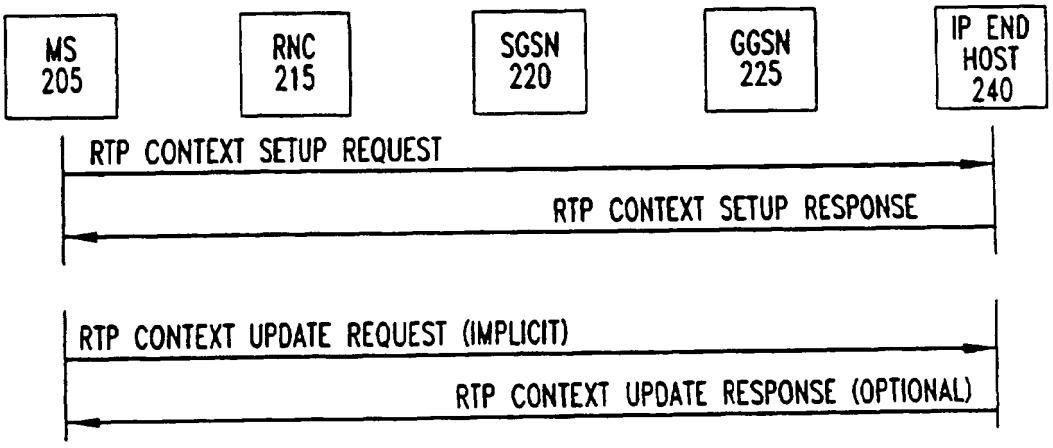


FIG. 9

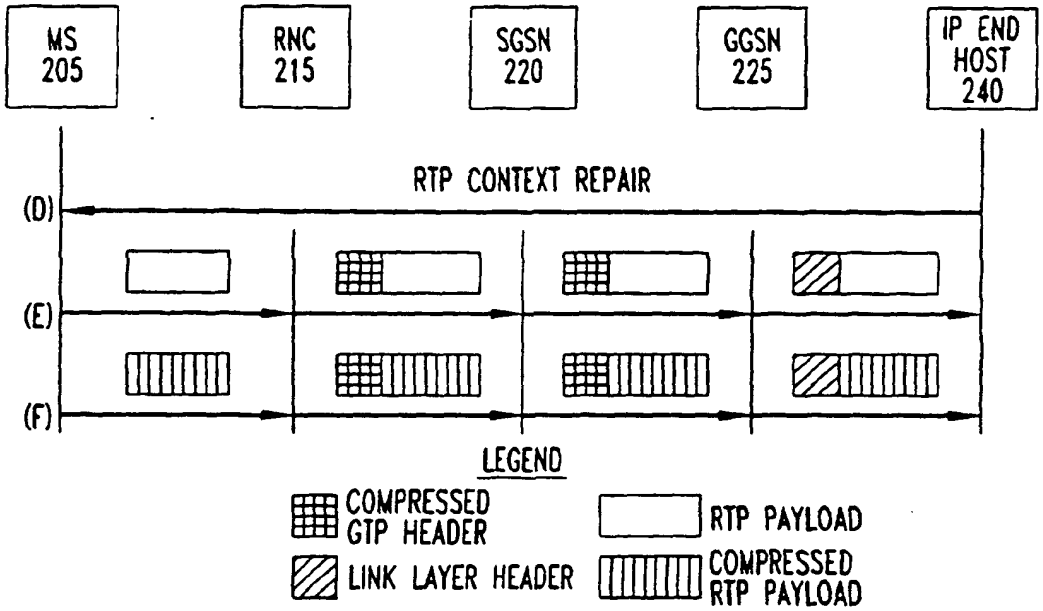
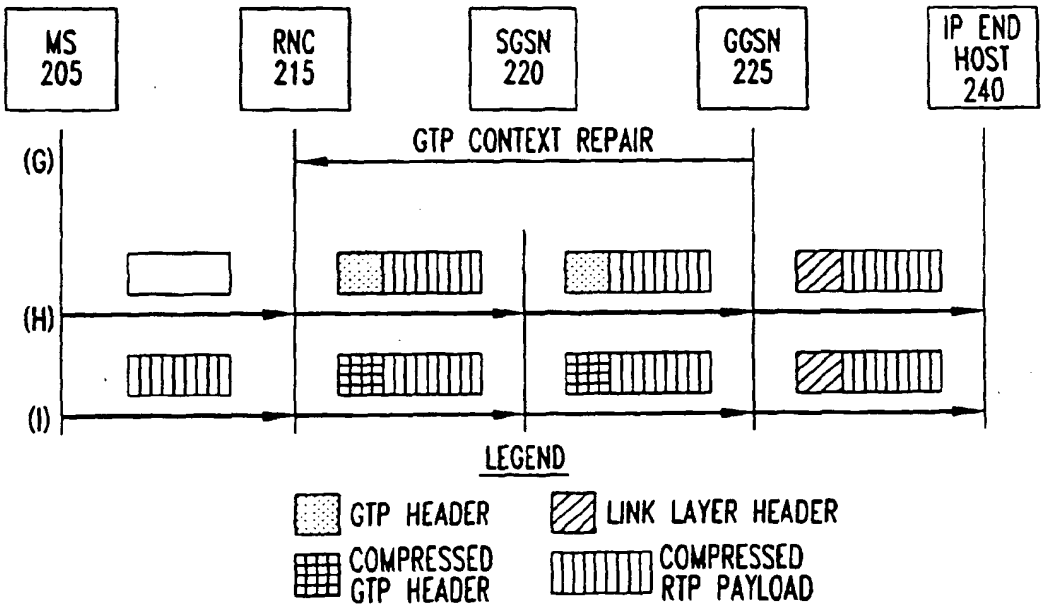


FIG. 10



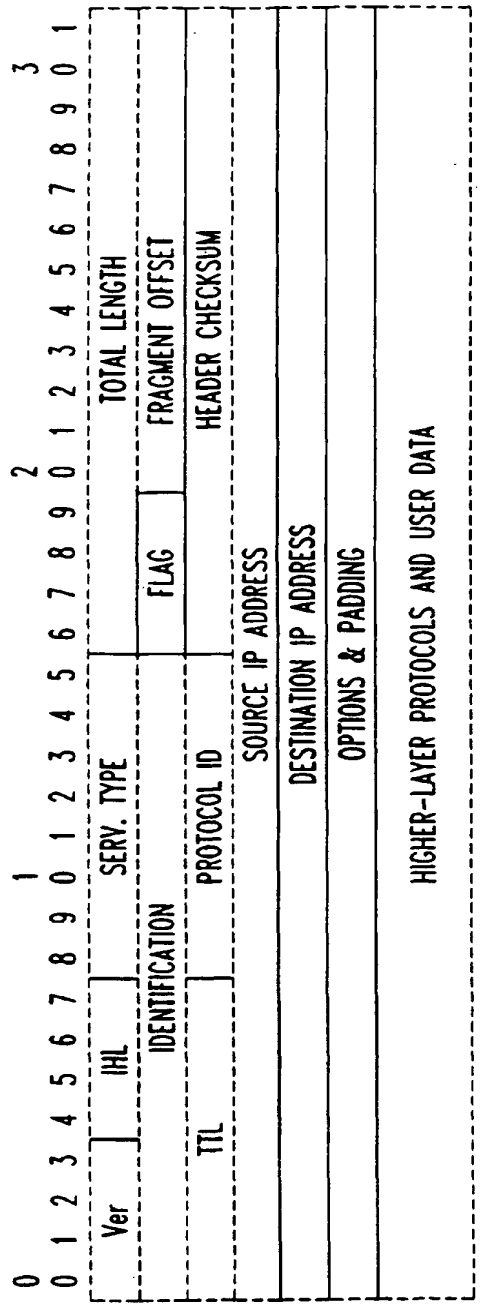


FIG. 11
(PRIOR ART)
IPv4 HEADER

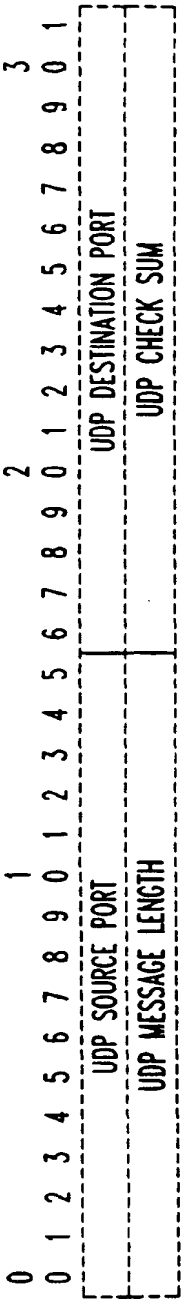


FIG. 12
(PRIOR ART)
UDP HEADER

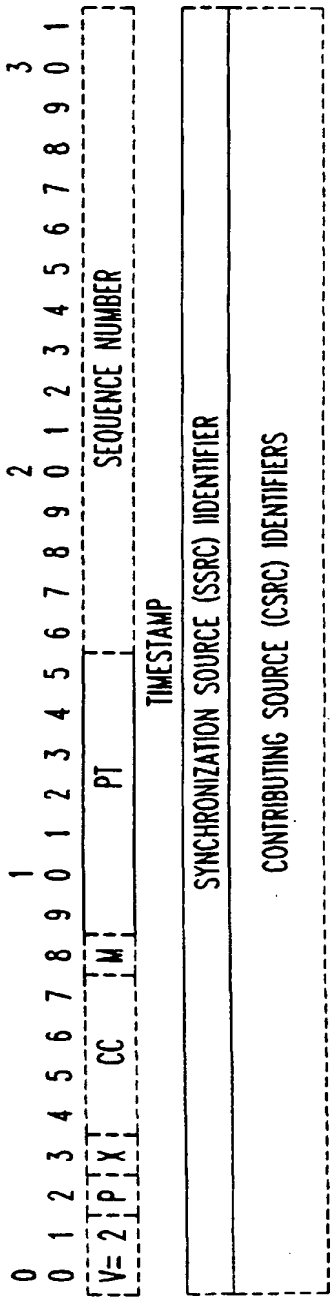


FIG. 13
(PRIOR ART)
RTP HEADER

FIG. 14

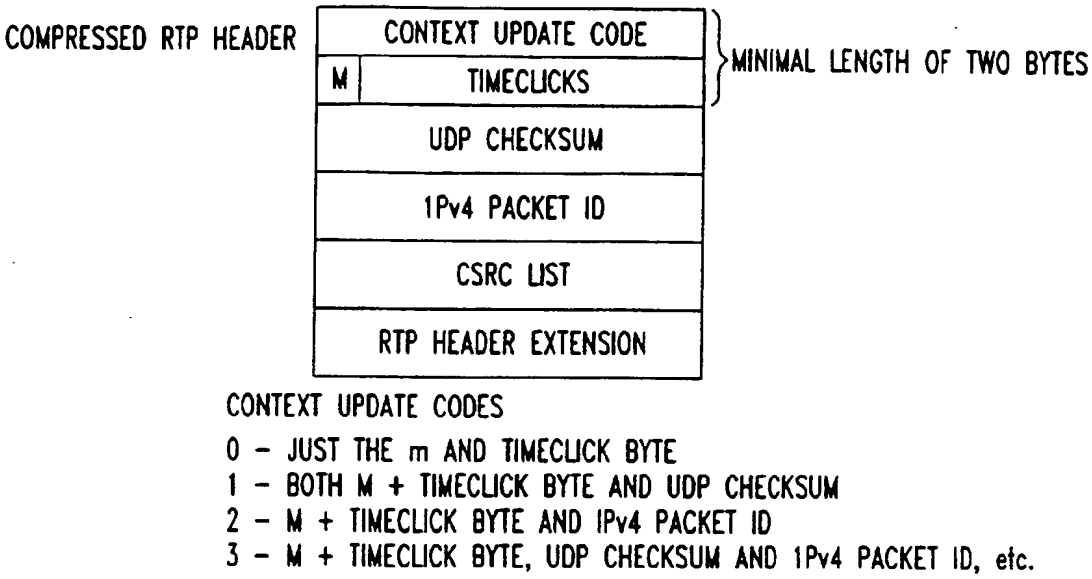


FIG. 15

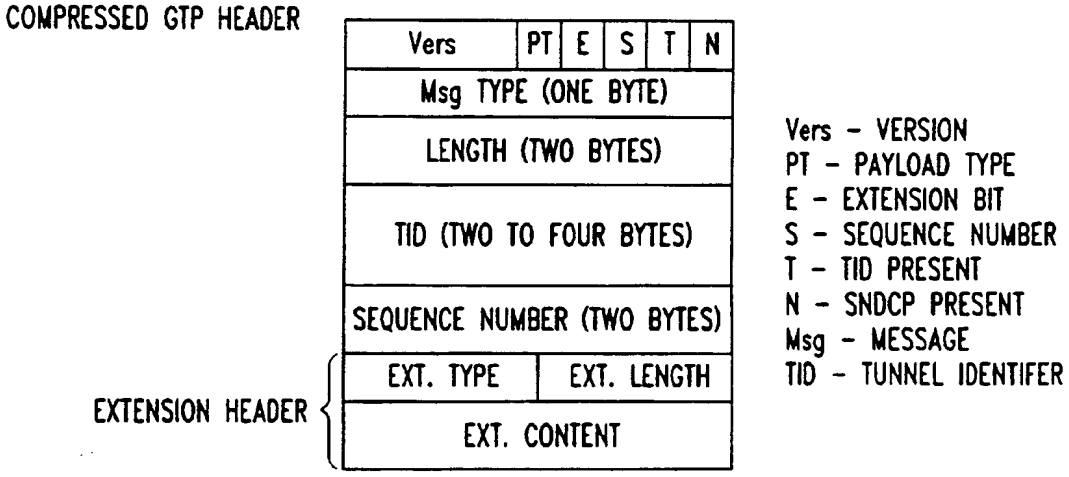
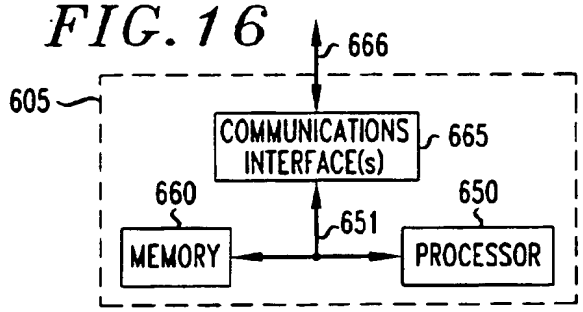


FIG. 16



European Patent Office

EUROPEAN SEARCH REPORT

Application Number
EP 00 30 7408

DOCUMENTS CONSIDERED TO BE RELEVANT

Category	Citation of document with indication, where appropriate, of relevant passages	Relevant to claim	CLASSIFICATION OF THE APPLICATION (Int.Cl.7)
A	W.DELYLLE: "GPRS and PDNs Interconnection Issues" , 'Online! August 1998 (1998-08), pages 1-78, XP002168897 Retrieved from the Internet: <URL:'PDF! www.ee.ucl.ac.uk/~lsacks/tcomsmsc/projects /pastproj/w_deylle.pdf> 'retrieved on 2001-06-06! * page 45, line 1 - page 48, last line * * page 55, line 1 - page 64, last line *	1,2,4	H04L29/06
A,D	CASNER S ET AL: "Compressing IP/UDP/RTP Headers for Low-Speed Serial Links" NETWORK WORKING GROUP; REQUEST FOR COMMENTS: 2508, February 1999 (1999-02), pages 1-24, XP002121319 * page 3, paragraph "3. The Compression Algorithm" - page 14, paragraph "3.3.4. Encoding of difference" *	1-4	TECHNICAL FIELDS SEARCHED (Int.Cl.7)
A	DEGERMARK M ET AL: "LOW-LOSS TCP/IP HEADER COMPRESSION FOR WIRELESS NETWORKS" WIRELESS NETWORKS,US,ACM, vol. 3, no. 5, 1 October 1997 (1997-10-01), pages 375-387, XP000728935 ISSN: 1022-0038 * page 376, right-hand column, line 1 - page 378, right-hand column, line 11 * * page 380, left-hand column, line 30 - right-hand column, line 6 *	1-4	H04L

The present search report has been drawn up for all claims

Place of search	Date of completion of the search	Examiner
THE HAGUE	8 June 2001	Behringer, L.V.

CATEGORY OF CITED DOCUMENTS

X : particularly relevant if taken alone
Y : particularly relevant if combined with another document of the same category
A : technological background
O : non-written disclosure
P : intermediate document

T : theory or principle underlying the invention
E : earlier patent document, but published on, or after the filing date
D : document cited in the application
L : document cited for other reasons

& : member of the same patent family, corresponding document

EPO FORM 1503 03.82 (P04C01)